Criteo AI LAB

Reward Optimizing Recommendation with Deep Learning and Maximum Inner Product Search

ECML PKDD Tutorial 2022

Imad Aouali, Amine Benhalloum, Martin Bompaire, Achraf Ait Sidi Hammou, Sergey Ivanov, Benjamin Heymann, David Rohde, Otmane Sakhi, Flavian Vasile, Maxime Vono

CRITEO

Agenda

1. **Introduction**
   Practice → Science
2. **Framework**
   Contextual Bandits
3. **Single Item Recommendation**
4. **Slate Recommendation**
5. **Conclusion**

Introduction

Practice

CRITEO

Introduction (Practice)

What are we trying to do?

Recommend the best items to a user

- Recommend: Within the context of online advertising, low latency constraints, slates
- Best: Maximizing a given reward (clicks, sales, …)
- Items: Tens of millions of products
- User: History of past user interactions, adapt to change in behaviour quickly, users enter and leave the system continually

Introduction (Practice)

General Framework

We are interested in decision rules of the form: $\text{score}(\boldsymbol{u}, a) \sim \boldsymbol{u}^T \boldsymbol{\beta}_a$

$\boldsymbol{u}$ – The user embedding of dimension D

$\boldsymbol{\beta}_a$ – The embedding of item a of dimension D

$\boldsymbol{\beta}$ – The matrix of item embeddings of dimension $D \times P$

Introduction (Practice)

General Framework

We are interested in decision rules of the form: $\text{score}(\boldsymbol{u}, a) \sim \boldsymbol{u}^T \boldsymbol{\beta}_a$

Why? To find the best action we need:

$$a^* = \text{argmax}_{a' \in \{1,..,P\}} \text{score}(\boldsymbol{u}, a')$$

If P is large, then this is very slow for general score functions.

Using Dot Product enables us to leverage (Approximate) Maximum Inner Product Search techniques.

Introduction (Practice)

General Framework

We are interested in decision rules of the form: $\text{score}(\boldsymbol{x}, a) \sim f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_a$

$\boldsymbol{x}$ – The context which is the information we use to personalize on.

f_{Ξ} - A mapping from context to user embedding.

Introduction (Practice)

General Framework

We are interested in decision rules of the form: $\text{score}(\boldsymbol{x}, a) \sim f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_a$

Why?

- Users enter and leave the system constantly.
- We want to adapt to change in behavior.
- Generalize across users.

Introduction (Practice)

General Framework

We are interested in decision rules of the form: $\text{score}(\boldsymbol{x}, a) \sim f_{\Xi}(\boldsymbol{x})^T h_{\boldsymbol{\beta}}(\boldsymbol{a})$

$\boldsymbol{a}$ – Set of initial pre-trained embeddings, product attributes and features …

Introduction (Practice)

General Framework

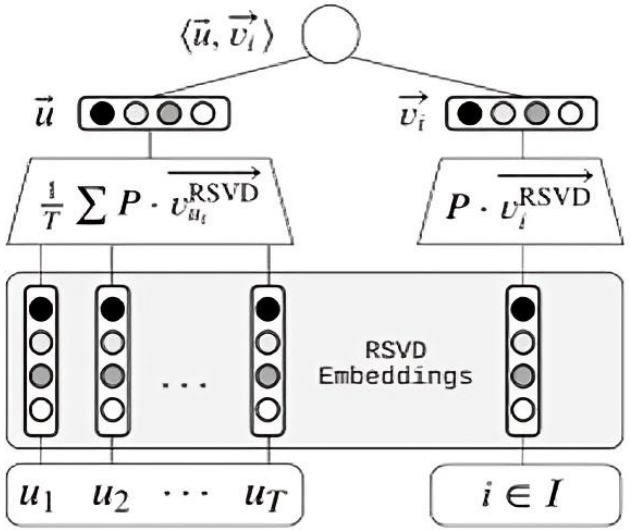
We are interested in decision rules of the form: $\text{score}(\boldsymbol{x}, a) \sim f_{\Xi}(\boldsymbol{x})^T h_{\boldsymbol{\beta}}(\boldsymbol{a})$

Why ?

- Learn from vast amounts of heterogeneous sources of data, side information, etc.
- Enable transfer between tasks.

Introduction (Practice)

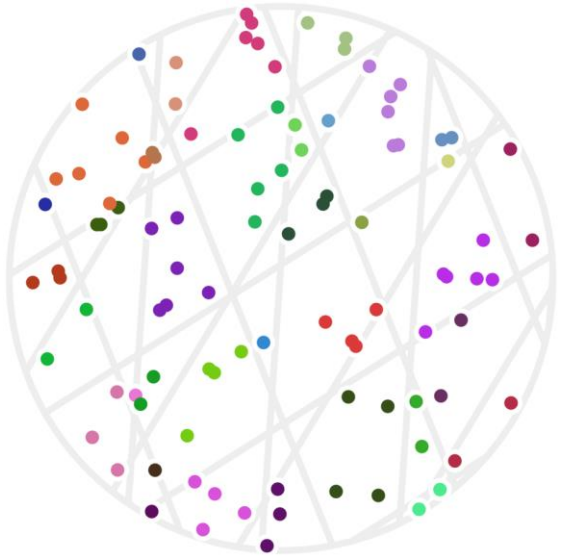
General Framework



- The now ubiquitous Two Tower style model (Criteo DeepKNN, Facebook DLRM ...)
- You can make the towers as shallow or deep as you want/need #deeplearning.

Introduction (Practice)

General Framework



$$\text{argsort}(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})_{\mathbf{1}:\boldsymbol{K}}$$

where

$$f_{\Xi}(\boldsymbol{x}) \; - \; D$$
$$\boldsymbol{\beta} \; - D \times P$$

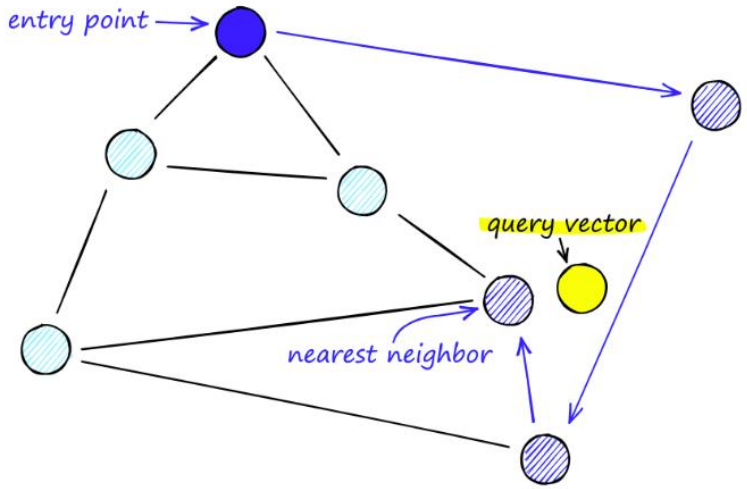
- Scores computation is $O(P \cdot D)$
- Exact sorting is $O(P \log P)$
- Exact top-K is $O(P)$

Solution: Approximate algorithms that "look like" $O(\log P)$

e.g., Navigable Small World

Introduction (Practice)

Side note: Hierarchical Navigable Small Worlds (HSNW)



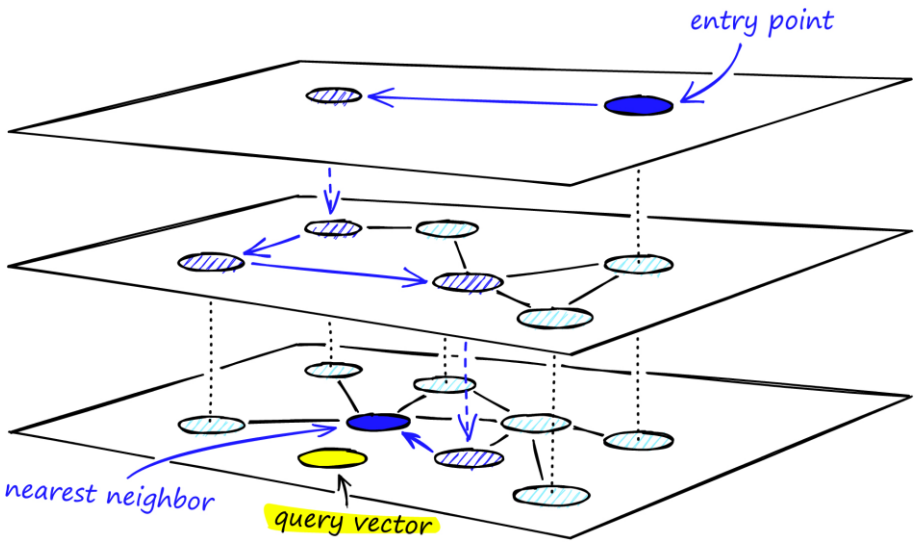
Credits: https://www.pinecone.io/learn/hnsw/

General recipe for Graph Based NN methods:

- Build a graph where close vectors are linked together.
- At query time, route your query vector through the graph greedily, navigating to the closest neighbor at each time.
- Stop when you hit a local minimum.

Introduction (Practice)

Side note: Hierarchical Navigable Small Worlds (HSNW)



Credits: https://www.pinecone.io/learn/hnsw/

The hierarchical part

- We start by a zoomed out view of the neighborhood graph
- And we zoom in after routing at each successive layer
- More fine grained search as we go deeper

Introduction (Practice)

Python Tutorial 1 (Exact vs. Approximate MIPS)

https://colab.research.google.com/github/otmhi/Reward-Optimizing-Reco/blob/main/ECML_Rewa[...]izing_Slate_Recommendation_with_DL_and_MIPS_Part_1.ipynb

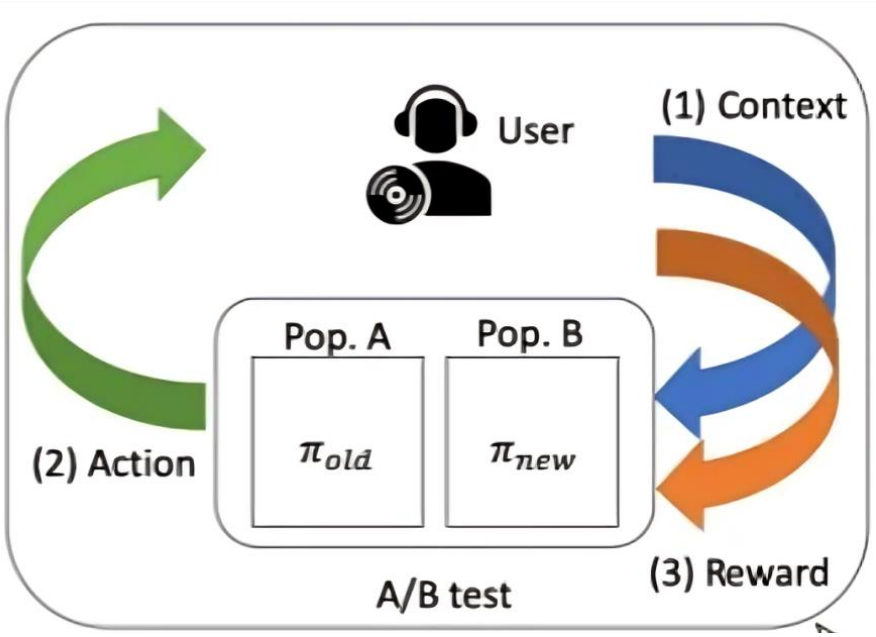
Introduction

Science

CRITEO

Introduction (Science)

Reward Optimizing Recommendation



Introduction (Science)

Reward Optimizing Recommendation

Population A

product viewed

recommended slate: $A($  $)$

product viewed

recommended slate: $A($  ,  $)$

$A(\cdot)$ is the algorithm we test on population A: it is a mapping from a list of viewed items to a list (slate) of recommended items.

Introduction (Science)

Reward Optimizing Recommendation

Population B

product viewed

product viewed

recommended slate: $B($  ,  $)$

product viewed

recommended s$($  $B($, , $)$.

$B(\cdot)$ is the algorithm we test on population B: it is a mapping from a list of viewed items to a list (or slate) of recommended items.

CRITEO

Introduction (Science)

Reward Optimizing Recommendation

Which is better $A(\cdot)$ or $B(\cdot)$?

- Recall that $A(\cdot)$ and $B(\cdot)$ are mappings from lists of items to lists of items e.g.

- In general, $A(\cdot)$ and $B(\cdot)$ are constrained by engineering limitations.
- An A/B test can measure the timeline reward accurately (If we make the stable unit treatment value assumption).

Introduction (Science)

Reward Optimizing Recommendation

How do we do this offline?

product viewed

product viewed

recommended slate: $B($  ,  $)$

product viewed

recommended slate: $B($  ,  ,  $)$

CRITEO

Introduction (Science)

Reward Optimizing Recommendation

How do we do this offline?

product viewed

product viewed

recommended slate: $B($  ,  $)$

product viewed

recommended slate: $B($  ,   $)$

→ It is really difficult !

CRITEO

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

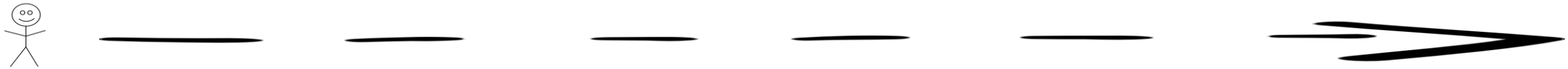
Our problem is that our recommender is a mapping:

And we're trying to answer the counterfactual question: what would happen if I had a different mapping? from data generated by a previous mapping where rewards could be delayed...

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?

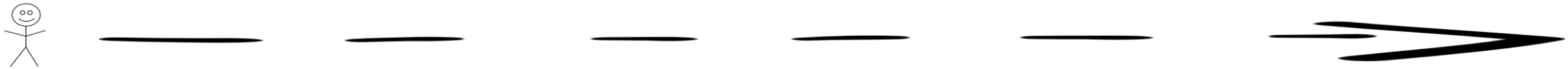


product viewed

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?



product viewed

product viewed

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?

product viewed

product viewed

product viewed

Becomes a positive example

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?

product viewed

product viewed

product viewed

$$L(\Xi, \boldsymbol{\beta}) = \log \sigma\{f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^+} - f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^-}\}$$

Becomes a positive example

CRITEO

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?

product viewed

product viewed

product viewed

$$L(\Xi, \boldsymbol{\beta}) = \log \sigma\{f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^+} - f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^-}\}$$

Becomes a positive example

The negative might be sampled uniformly

CRITEO

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How we do this offline?

product viewed

product viewed

product viewed

$$L(\Xi, \boldsymbol{\beta}) = \log \sigma\{f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^+} - f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^-}\}$$

Becomes a positive example

The negative might be sampled uniformly or using popularity

CRITEO

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How do we use recommendation feedback ?

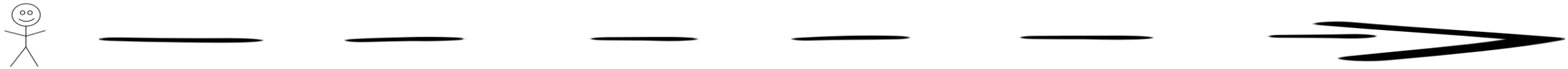
First answer, use a ranking loss function while training on historical recommendations and feedback.

e.g., Some pairwise ranking loss such as BPR

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How do we use recommendation feedback ?

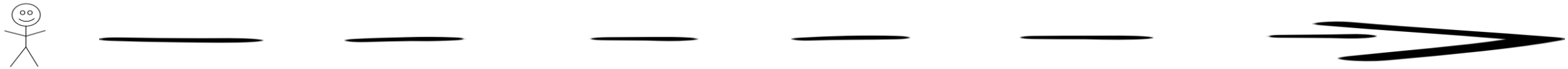


product
viewed

Introduction (Science)

Non-Reward Optimizing Recommendation with Pseudo Rewards

How do we use recommendation feedback ?



product viewed

product viewed

Introduction (Science)

Reward Optimizing Recommendation

How do we use recommendation feedback ?

product viewed

product viewed

recommended slate:

No click click No click

CRITEO

Introduction (Science)

Reward Optimizing Recommendation

How do we use recommendation feedback ?

product viewed

product viewed

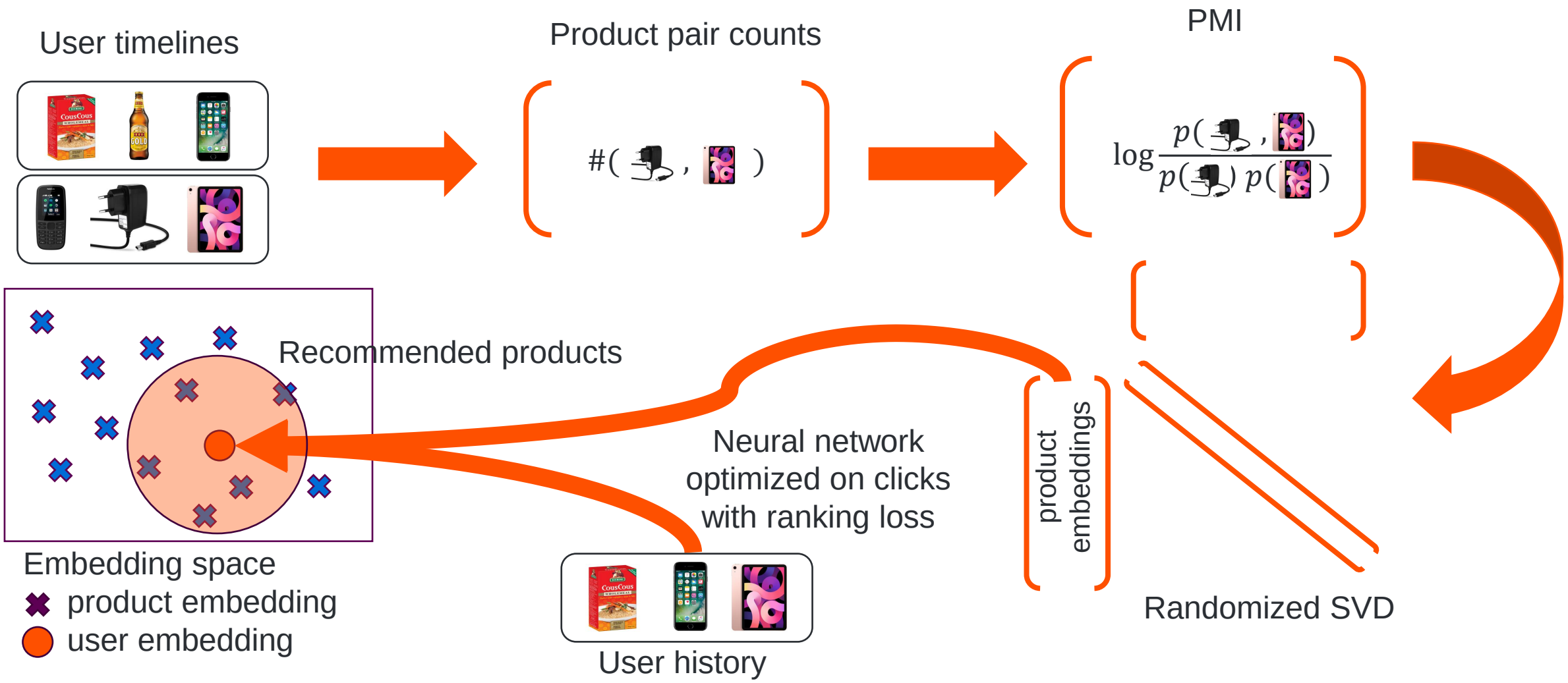
recommended slate:

No click click No click

$$L(\Xi, \boldsymbol{\beta}) = \log \sigma\{f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^+} - f_{\Xi}(v_1 = \text{  }, v_2 = \text{  })^T \boldsymbol{\beta}_{a^-}\}$$

CRITEO

Recommendation at Criteo



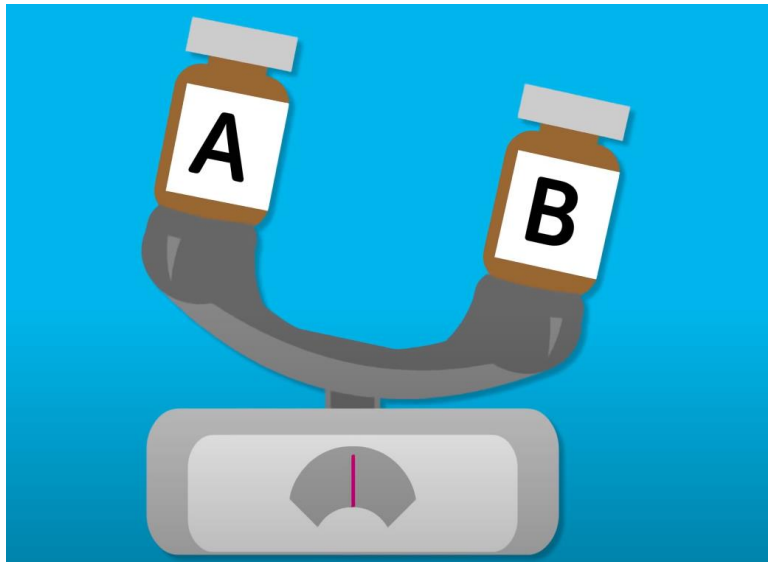
Introduction

Reward on the Timeline

CRITEO

Introduction (Reward on the Timeline)

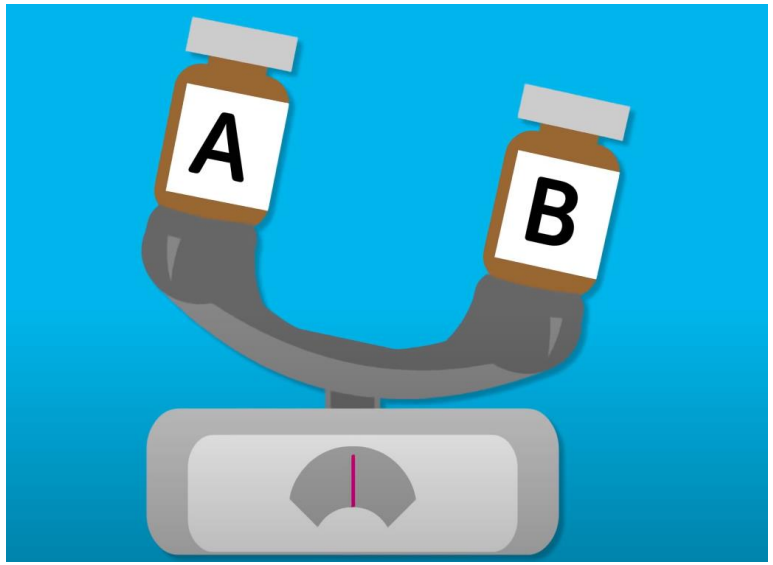
Randomized Control Trials (RCT) in Medicine



In medicine a randomized control trial, splits the population into two groups.

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

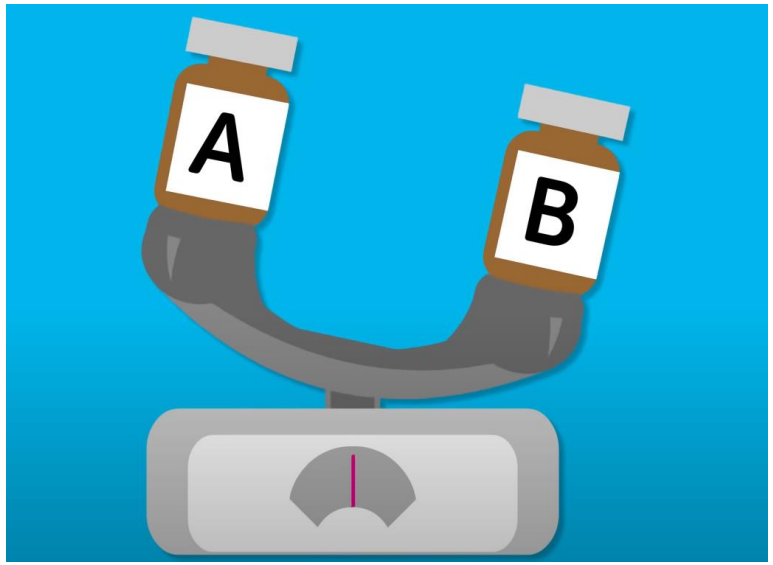


In medicine a randomized control trial, splits the population into two groups. Here, we have

- A potentially strong intervention.

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

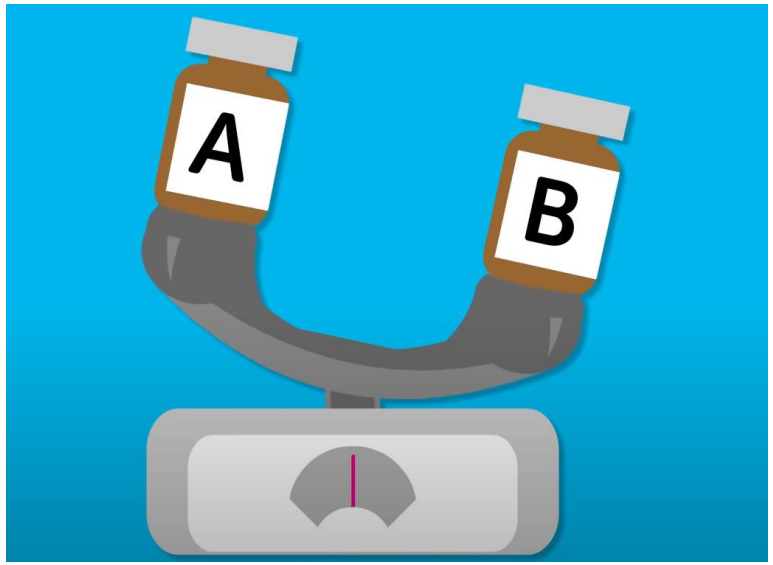


In medicine a randomized control trial, splits the population into two groups. Here, we have

- A potentially strong intervention.
- Little personalization (clear membership).

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

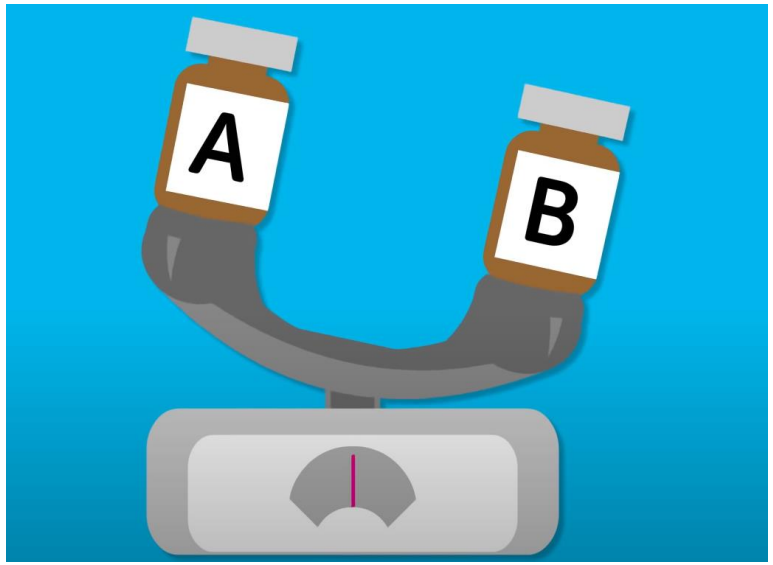


In medicine a randomized control trial, splits the population into two groups. Here, we have

- A potentially strong intervention.
- Little personalization (clear membership).
- Usually just two actions.

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

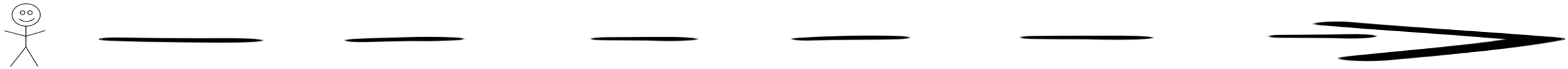


In medicine a randomized control trial, splits the population into two groups. Here, we have

- A potentially strong intervention.
- Little personalization (clear membership).
- Usually just two actions.
- Lots of negative results!

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine



product viewed

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

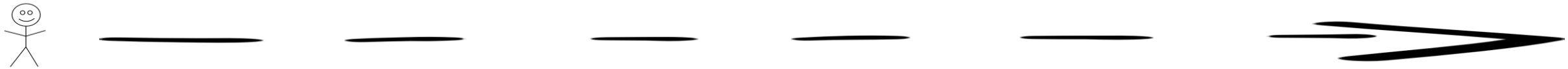
product viewed

product viewed

CRITEO

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine



product viewed

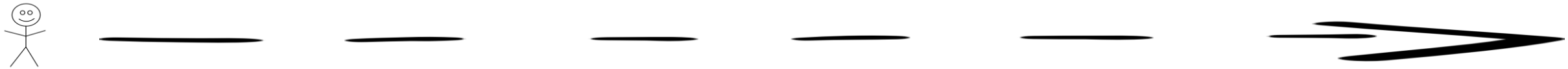
product viewed

recommended slate:

CRITEO

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine



product viewed

product viewed

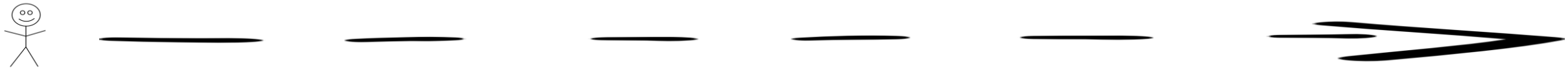
recommended slate:

No click

CRITEO

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine



product viewed

product viewed

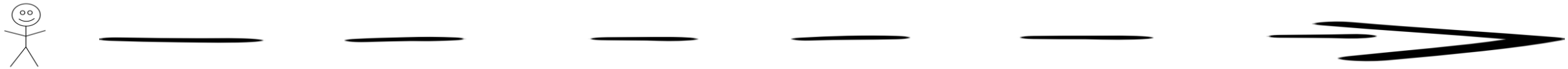
recommended slate:

product viewed

CRITEO

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine



product viewed

product viewed

recommended slate:

product viewed

recommended slate:

CRITEO

Introduction (Reward on the Timeline)

Randomized Control Trials (RCT) in Medicine

product viewed

product viewed

recommended slate:

product viewed

recommended slate:

No click

CRITEO

Introduction (Reward on the Timeline)

RCT in Medicine vs. A/B Tests in Recommendation

	RCT in Medicine	A/B Tests in Recommendation
Personalization segments	• The user is who they are. They are categorized by a small number of attributes e.g., age band or sex.	• The User is what they do. If they view an item – then that is who they are and what we want to personalize on.
Size of personalization segments	• Small: age bands, sex, simple medical history	• Complex – everything we know so far about the user.

Introduction (Reward on the Timeline)

RCT in Medicine vs. A/B Tests in Recommendation

	RCT in Medicine	A/B Tests in Recommendation
Action size	• Often just intervention and placebo – (dose and dosing strategy).	• With a single item – very big. With a list (or slate) of items – the scale of the number of particles in the universe.
Effect size	• Large if it works.	• Usually relatively small.

Introduction (Reward on the Timeline)

Reinforcement Learning for Games vs Recommendation

Games: In certain very successful RL settings, the problem is both similar and different to recommendation. Here, we have:

- No cost to losing many times before you win.
- Strong reward signals with little noise.
- Sequential steps of actions are critical to unlocking reward.

Introduction (Reward on the Timeline)

Reinforcement Learning for Games vs Recommendation

Recommendation: In principle it is the right formulation to move reward to the timeline. Here, we have

- High cost to performing poor actions in the wild.
- Lots of noise.
- Modest episodic features.

Introduction

Decision Theory for Recommendation

CRITEO

Introduction (Decision Theory for Recommendation)

Example: Leonard Jimmie Savage in 1951

What are the states of nature that might occur?

→ Rain / No Rain

CRITEO

Introduction (Decision Theory for Recommendation)

Example: Leonard Jimmie Savage in 1951

What are the actions that we might take?

→ Carry umbrella / Leave umbrella

CRITEO

Introduction (Decision Theory for Recommendation)

Example: Leonard Jimmie Savage in 1951

Act \ State	Rain	Shine
Carry	Inconvenience and wet feet	Inconvenience and slight embarrassment
Don't carry	Miserable drenching	Bliss unalloyed

CRITEO

Introduction (Decision Theory for Recommendation)

Example: Leonard Jimmie Savage in 1951

State / Act	Rain	Shine
Carry	Inconvenience and wet feet	Inconvenience and slight embarrassment
Don't carry	Miserable drenching	Bliss unalloyed

CRITEO

Introduction (Decision Theory for Recommendation)

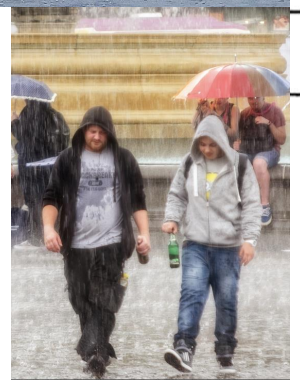
Example: Leonard Jimmie Savage in 1951

State / Act	Rain	Shine
Carry	Inconvenience and wet feet	Inconvenience and slight embarrassment
Don't carry	Miserable drenching	Bliss unalloyed

CRITEO

Introduction (Decision Theory for Recommendation)

Example: Leonard Jimmie Savage in 1951



State / Act	Rain	Shine
Carry	Inconvenience and wet feet	Inconvenience and slight embarrassment
Don't carry	Miserable drenching	Bliss unalloyed

CRITEO

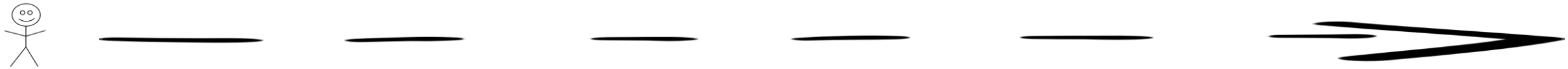
Introduction (Decision Theory for Recommendation)

In the Recommendation Case:

- What are the states of nature? (rain or no rain)
- What are the actions? (bring umbrella / leave umbrella)

Introduction (Decision Theory for Recommendation)

States of Nature and Decision

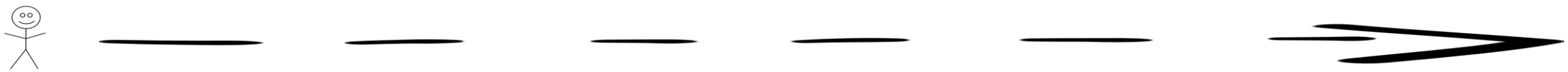


product viewed

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision



product viewed

product viewed

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision

product viewed

product viewed

recommended slate:

?

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision

product viewed

product viewed

recommended slate:

?

No click

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision

product viewed

product viewed

recommended slate:

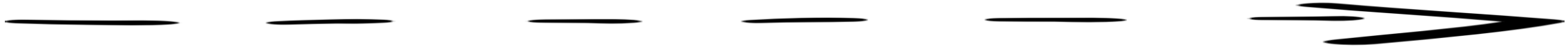
?

product viewed

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision



product viewed

product viewed

recommended slate:

?

product viewed

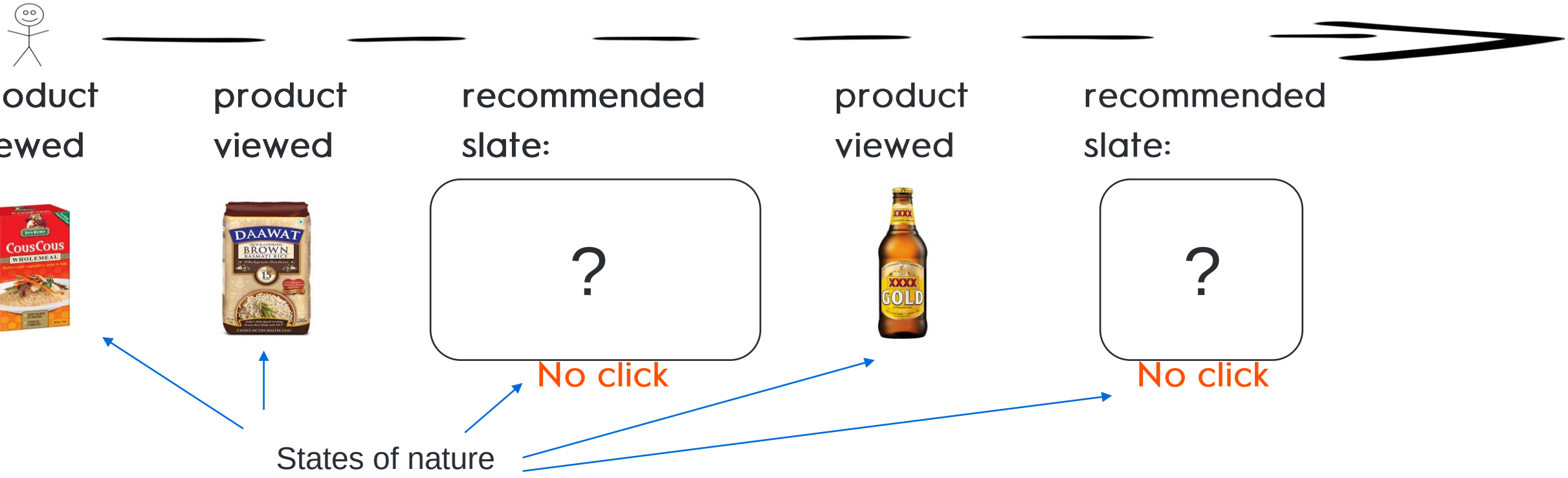
recommended slate:

?

CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision



CRITEO

Introduction (Decision Theory for Recommendation)

States of Nature and Decision

product viewed

product viewed

recommended slate:

?

No click

product viewed

recommended slate:

?

No click

Slate = $\text{argsort}(f(v_1 =$  $v_2 =$  $)^T \boldsymbol{\beta})_{1:3}$

Decision Rule

Slate = $\text{argsort}(f(v_1 =$  $v_2 =$  $v_3 =$  $)^T \boldsymbol{\beta})_{1:}$

CRITEO

Introduction (Decision Theory for Recommendation)

In the Recommendation Case:

- What are the states of nature? Complex timelines of user behavior
- What are the actions? A decision rule that we can implement in production e.g., a MIPS search

The benefit of decision theory is we can restrict the complexity of the production system (decision rule) – without simplifying our understanding of the real world.

For example, we can measure the utility of two MIPS A and B by A/B test.

It is only possible to evaluate to candidate recommenders (driven by MIPS A and MIPS B) by understanding the probable states of nature the recommender will face. Doing this without an A/B test requires *off-policy evaluation*.

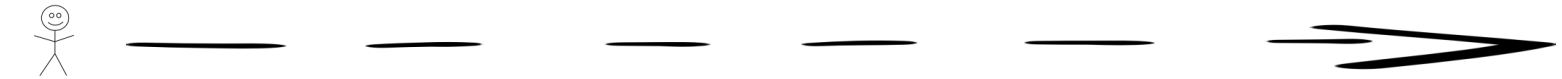
Framework

Contextual Bandits

CRITEO

Framework (Contextual Bandits)

Recall the full recommendation problem has the following form:



The story is:

1. *We learn something,*
2. The recommender system does something,
3. *We learn something new,*
4. … and so on …

Eventually we observe some reward

Framework (Contextual Bandits)

Recall the full recommendation problem has the following form:

The story is:

1. *We learn something,*
2. The recommender system does something,
3. *We learn something new,*
4. … and so on …

Eventually we observe some reward : this is complicated and except at A/B test time we do not know how to produce a reasonable *off policy estimate of reward.*

CRITEO

Framework (Contextual Bandits)

A simplified problem

The simplified story is:

1. *We learn something,*
2. The recommender system does something,
3. *We observe reward (click)*

We take slates (not users) to be conditionally independent. *Not epidemiologically correct.*

Decision Theory for Contextual Bandits

Decision theory for contextual bandits takes a particularly simplified form

$\boldsymbol{x}$ - context

R – reward

Decision rule $B(\cdot)$

Distribution on contexts $\nu(\boldsymbol{x})$

Conditional distribution on reward $R(\boldsymbol{x}, a) = P[R = 1|\boldsymbol{x}, a]$

The expected reward V is then given by:

$$V\big(B(\cdot)\big) = \mathbb{E}_{x\sim\nu}[R\big(\boldsymbol{x}, B(\boldsymbol{x})\big)]$$

Decision Theory for Contextual Bandits

Decision theory for contextual bandits takes a particularly simplified form

$\boldsymbol{x}$ - context

R – reward

Decision rule $B(\cdot)$ - we might want to constrain this to be a MIPS to be consistent with engineering constraints i.e.

$$B(\boldsymbol{x}) = \text{argmax}(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})$$

Distribution on contexts $\nu(x)$

Conditional distribution on reward $R(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a)$

The expected reward V is then given by:

$$V(f_{\Xi}(\cdot), \boldsymbol{\beta}) = \mathbb{E}_{x\sim\nu}[R\big(\boldsymbol{x}, \text{argmax}(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})\big)]$$

Decision Theory for Contextual Bandits

Decision theory for contextual bandits takes a particularly simplified form

$\boldsymbol{x}$ - context

R – reward

Decision rule $B(\cdot)$ - we might further relax the decision rule to a policy

$$\pi(a|x) = \frac{\exp(f_{\Xi}(x)^T \beta_a)}{\sum_{a'} \exp(f_{\Xi}(x)^T \beta_{a'})}$$

Distribution on contexts $\nu(x)$

Conditional distribution on reward $R(x, a) = P(R = 1|x, a)$

The expected reward V is then given by:

$$V(f_{\Xi}(\cdot), \beta) = \mathbb{E}_{x \sim \nu} \mathbb{E}_{a \sim \pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a)]$$

Decision Theory for Contextual Bandits

Decision theory for contextual bandits takes a particularly simplified form

$\boldsymbol{x}$ - context

R – reward

Decision rule $B(\cdot)$ - we might further relax the decision rule to a policy

$$\pi(a|x) = \frac{\exp(f_{\Xi}(x)^T \beta_a)}{\sum_{a'} \exp(f_{\Xi}(x)^T \beta_{a'})}$$

Distribution on contexts $\nu(x)$

Conditional distribution on reward $R(x, a) = P(R = 1|x, a)$

The expected reward V is then given by:

$$V(f_{\Xi}(\cdot), \beta) = V(\pi) = \mathbb{E}_{x\sim\nu}\mathbb{E}_{a\sim\pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a)]$$

Our fundamental problem is to find the optimal MIPS i.e., $f_{\Xi}(\cdot), \beta$. In order to evaluate this, we need to understand how the world works i.e., $\nu(x)$ and particularly $P(R = 1|x, a) = R(x, a)$.

$R(x, a)$ is particularly important because the best a^* for a given x is given by:

$$\mathrm{a}^* = \mathrm{argmax}_{\mathrm{a}} R(a, x)$$

Decision Theory for Contextual Bandits

Decision theory for contextual bandits takes a particularly simplified form

$\boldsymbol{x}$ - context

R – reward

Decision rule $B(\cdot)$ - we might further relax the decision rule to a policy

$$\pi(a|x) = \frac{\exp(f_{\Xi}(x)^T \beta_a)}{\sum_{a'} \exp(f_{\Xi}(x)^T \beta_{a'})}$$

Distribution on contexts $\nu(x)$

Conditional distribution on reward $R(x, a) = P(R = 1|x, a)$

The expected reward V is then given by:

$$V(f_{\Xi}(\cdot), \beta) = V(\pi) = \mathbb{E}_{x\sim\nu}\mathbb{E}_{a\sim\pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a)]$$

Key reward optimizing recommendation approaches focus on:

- Off policy estimation of $R(x, a)$.
- Given access to an estimate of $R(x, a)$ how do we get the decision rule $f_{\Xi}(\cdot), \beta$.

Successful reward optimizing recommendation requires both.

Framework

Off-Policy Reward Estimation

CRITEO

Framework (Off-Policy Reward Estimation)

Two Families of Approaches

- The goal of a reward optimizing recommender system is to find a policy π that maximizes the value function

$$V(\pi) = \mathbb{E}_{x\sim\nu}\mathbb{E}_{a\sim\pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a)]$$

- We never have access to the true value function $V(\pi)$, thus the first step is to find an estimate $\hat{V}_n(\pi)$ of $V(\pi)$. There are two families of approaches for this purpose, Direct Methods and Inverse Propensity Scoring.

Framework (Off-Policy Reward Estimation)

Two Families of Approaches

Direct Methods (DM): Find a model-based estimator $\hat{R}^{DM}(\boldsymbol{x}, a)$ of the expected reward $R(\boldsymbol{x}, a)$. The estimator of the value function follows as

$$\hat{V}_n^{DM}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_{a}\hat{R}^{DM}(\boldsymbol{x}_i, a)\pi(a \mid \boldsymbol{x}_i)$$

Inverse Propensity Scoring (IPS): Design unbiased estimators of $V(\pi)$ by re-weighting samples using the importance sampling trick

$$\hat{V}_n^{IPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n} R_i \frac{\pi(a_i \mid \boldsymbol{x}_i)}{\pi_0(a_i \mid \boldsymbol{x}_i)}$$

Framework (Off-Policy Reward Estimation)

Two Families of Approaches

Direct Methods (DM): $\hat{V}_n^{DM}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_a \hat{R}^{DM}(\boldsymbol{x}_i, a)\pi(a \mid \boldsymbol{x}_i)$

Inverse Propensity Scoring (IPS): $\hat{V}_n^{IPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n} R_i \frac{\pi(a_i|\boldsymbol{x}_i)}{\pi_0(a_i|\boldsymbol{x}_i)}$

→ IPS can be seen as a DM where :

$$\hat{R}^{\mathrm{IPS}}(\mathrm{x},\mathrm{a}) = \begin{cases} \dfrac{1}{|\mathcal{I}_n(\mathrm{x},\mathrm{a})|}\displaystyle\sum_{\mathrm{i}\in\mathcal{I}_n(\mathrm{x},\mathrm{a})}\dfrac{\mathrm{R_i}}{\pi_0(\mathrm{a_i}|\mathrm{x_i})} & \text{if } |\mathcal{I}_n(\mathrm{x},\mathrm{a})| \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

where $\mathcal{I}_n(x,a) = \{i \in [n]\,,\ x_i = x \text{ and } a_i = a\}$

Framework (Off-Policy Reward Estimation)

Two Families of Approaches

Direct Methods (DM): $\hat{V}_n^{DM}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_a \hat{R}^{DM}(\boldsymbol{x}_i, a)\pi(a \mid \boldsymbol{x}_i)$

Inverse Propensity Scoring (IPS): $\hat{V}_n^{IPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n} R_i \frac{\pi(a_i|\boldsymbol{x}_i)}{\pi_0(a_i|\boldsymbol{x}_i)}$

→ $\hat{R}(\boldsymbol{x}_i, a)$ will denote an estimator of the expected reward $R(x, a)$ using either DM or IPS.

→ Similarly, we refer by $\hat{V}_n(\pi)$ an estimator of the value function using either $\hat{V}_n^{DM}(\pi)$ or $\hat{V}_n^{IPS}(\pi)$.

$$\hat{V}_n(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_a \hat{R}\,(\boldsymbol{x}_i, a)\pi(a \mid \boldsymbol{x}_i)$$

Single Item Recommendation

IPS Method

CRITEO

Single Item Recommendation (IPS Method)

IPS for Single Items

- $\hat{V}_n^{IPS}(\pi)$ is provably unbiased under mild assumptions.

- However, the variance of $\hat{V}_N^{IPS}(\pi)$ grows linearly with the importance weight $\frac{\pi(a_i|\boldsymbol{x}_i)}{\pi_0(a_i|\boldsymbol{x}_i)}$.

  → the variance explodes when π drifts away from π_0.

- **Example**: The logging policy π_0 is uniform. In this case, the variance grows with $P\pi(a|x)$, where P is the catalog size (massive for large catalogs).

Single Item Recommendation (IPS Method)

IPS for Single Items

- Existing solutions of the variance problem of $\hat{V}_n^{IPS}(\pi)$ include:
  - Clipping the importance weights: replacing $\frac{\pi(a|x)}{\pi_0(a|x)}$ by $\min\left(\frac{\pi(a|x)}{\pi_0(a|x)}, M\right)$.
  - Self normalization of the importance weights.
  - Combining DM and IPS using doubly robust estimation (DR).

  → Introduce a bias-variance tradeoff but still difficult when P is large.

Dudík, Miroslav, et al. 2012.

Bottou, Léon, et al. 2013

Swaminathan, Adith, and Thorsten Joachims. 2015

Single Item Recommendation

Direct Method

CRITEO

Single Item Recommendation (Direct Method)

Logistic Regression Model

$$R(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(g_{\Gamma}(\boldsymbol{x})^T \boldsymbol{\Psi}_a)$$

- Can be biased but doesn't suffer from a variance problem.
- Reduce the bias by increasing the embeddings dimension D.

BLOB – uses priors to accelerate learning the reward in large action spaces.

Because of the parameterization of the reward model, we automatically get a MIPS:

$$a^* = \text{argmax}_a(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta_a})$$

with $f_{\Xi}(.) \leftarrow g_{\Gamma}(.)$ and $\boldsymbol{\beta} \leftarrow \boldsymbol{\Psi}$ ← it can result in larger than necessary embeddings

Single Item Recommendation (Direct Method)

Logistic Regression Model

$$R(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(g_{\Gamma}(\boldsymbol{x})^T \boldsymbol{\Psi_a})$$

- Can be biased but doesn't suffer from a variance problem.
- Reduce the bias by increasing the embeddings dimension D.

BLOB – uses priors to accelerate learning the reward in large action spaces.

Because of the parameterization of the reward model, we automatically get a MIPS:

$$a^* = \text{argmax}_a(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta_a})$$

with $f_{\Xi}(.) \leftarrow g_{\Gamma}(.)$ and $\boldsymbol{\beta} \leftarrow \boldsymbol{\Psi}$

We will discuss alternative ways to get $f_{\Xi}(\cdot), \boldsymbol{\beta}$

Single Item Recommendation (Direct Method)

Logistic Regression Model

We can split $\boldsymbol{x} = [\boldsymbol{y}, \boldsymbol{z}]$ where

$\boldsymbol{y}$ – User Engagement, and $\boldsymbol{z}$ – User interest

$$R(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(\boldsymbol{y}^T \boldsymbol{\phi} + g_\Gamma(\boldsymbol{z})^T \boldsymbol{\Psi}_{\boldsymbol{a}})$$

$$a^* = \text{argmax}_a (f_\Xi(\boldsymbol{z})^T \boldsymbol{\beta}_{\boldsymbol{a}})$$

with $f_\Xi(.) \leftarrow g_\Gamma(.)$ and $\boldsymbol{\beta} \leftarrow \boldsymbol{\Psi}$

Single Item Recommendation (Direct Method)

Logistic Regression Model

We can split $\boldsymbol{x} = [\boldsymbol{y}, \boldsymbol{z}]$ where

$\boldsymbol{y}$ – User Engagement, and $\boldsymbol{z}$ – User interest

$$R(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(\boldsymbol{y}^T \boldsymbol{\phi} + g_\Gamma(\boldsymbol{z})^T \boldsymbol{\Psi_a})$$

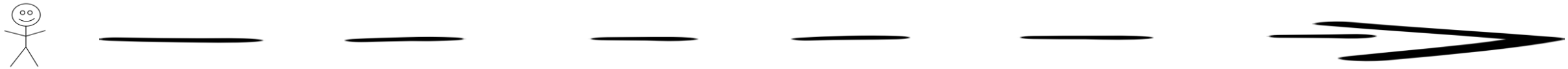
$$a^* = \text{argmax}_a (f_\Xi(\boldsymbol{z})^T \boldsymbol{\beta_a})$$

Nuisance parameters

with $f_\Xi(.) \leftarrow g_\Gamma(.)$ and $\boldsymbol{\beta} \leftarrow \boldsymbol{\Psi}$

Single Item Recommendation (Direct Method)

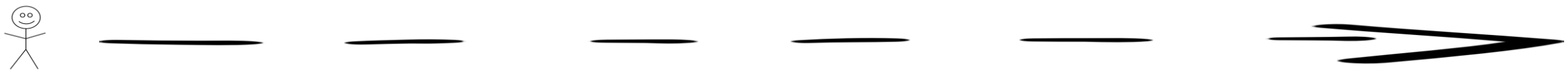
Remarks



product viewed

Single Item Recommendation (Direct Method)

Remarks



product viewed

product viewed

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

Click/No click

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

Click/No click

Recommend = $\text{argmax}(f(v_1 =$  $, v_2 =$  $)^T \boldsymbol{\beta})$

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

Click/No click

Recommend = argmax($f(v_1 =$  $, v_2 =$  $)^T \boldsymbol{\beta}$)

How do we build the reward model?

We can also use features like:

- How active is the user?
- How big/attractive is the recommendation?

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

Click/No click

Recommend = argmax($f(v_1 =$  $, v_2 =$  $)^T \boldsymbol{\beta}$)

How do we build the reward model?

We can also use features like:
- How active is the user?
- How big/attractive is the recommendation?

This (often) dominates in practice

CRITEO

Single Item Recommendation (Direct Method)

Remarks

product viewed

product viewed

recommended ~~slate~~ item:

Click/No click

Recommend = $\text{argmax}(f(v_1 = $  $, v_2 = $  $)^T \boldsymbol{\beta})$

How do we build the reward model?

We can also use features like:
- How active is the user?
- How big/attractive is the recommendation?

We can predict well using only these features

CRITEO

Single Item Recommendation

Decision Rule Optimization

CRITEO

Single Item Recommendation (Decision Rule Optimization)

Why Optimize $\boldsymbol{\pi(a \mid x)}$ when $R(\boldsymbol{x}, a)$ is a Model?

If our reward model has the following parametric form:

$$\hat{R}(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(g_{\Gamma}(\boldsymbol{x})^T \boldsymbol{\Psi}_a)$$

Then we already have an optimal MIPS: $f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}$, by setting

$$f_{\Xi}(\boldsymbol{x}) \leftarrow g_{\Gamma}(\boldsymbol{x}), \qquad \boldsymbol{\beta} \leftarrow \boldsymbol{\Psi}$$

So why would we optimize: $\hat{V}_n(\pi) = \sum_{i=1}^{n} \sum_a \hat{R}\,(\boldsymbol{x}_i, a)\pi(a \mid \boldsymbol{x}_i)$ to obtain a MIPS if we already have one?

→$\hat{R}(\boldsymbol{x}, a) = P(R = 1|\boldsymbol{x}, a) = \sigma(g_{\Gamma}(\boldsymbol{x})^T \boldsymbol{\Psi}_a)$ is sufficient but not necessary.

→A few reasons we might want not to use it in the decision rule.

Single Item Recommendation (Decision Rule Optimization)

Why Optimize $\boldsymbol{\pi(a \mid x)}$ When $R(\boldsymbol{x}, a)$ is a Model?

→ A few reasons we might want not to use it in the decision rule

- $\sigma(\boldsymbol{y}^T \boldsymbol{\phi} + f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_a)$ is a calibrated reward that correctly orders all actions. We only need the best action.
- Imposing the above can mean the embedding dimension D for $f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}$ might be larger than we need.
- We might want the dimension of $\boldsymbol{\beta}$ lower than D for engineering constraints.
- We might prefer a more general model $P(R = 1|a, \boldsymbol{x}) = \sigma\big(h_{\Phi}(\boldsymbol{x}, a)\big)$.

Single Item Recommendation (Decision Rule Optimization)

Optimizing the decision rule via a policy

The objective is:

$$\hat{V}_n(\pi_{\Xi,\beta}) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}[\hat{R}(a,\boldsymbol{x})]$$

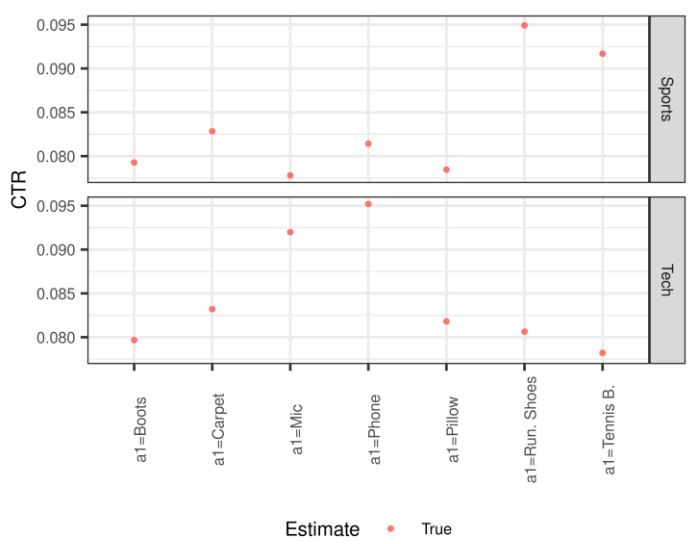
where :

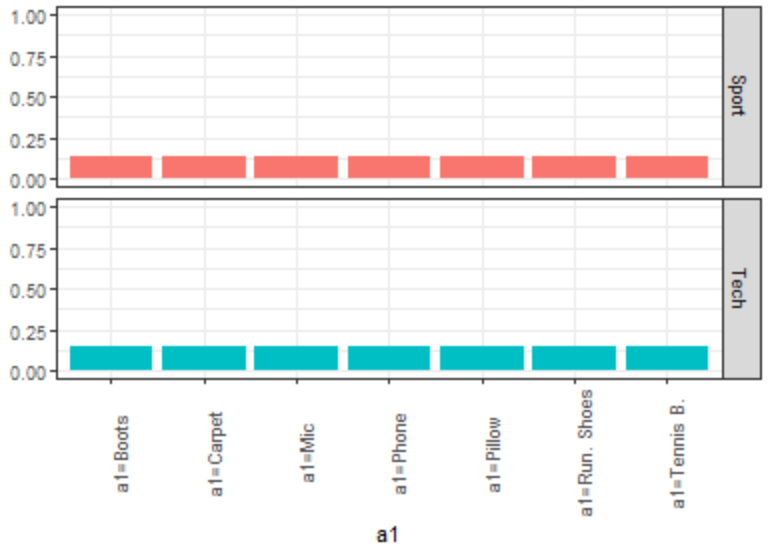
$$\pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_a)}{\sum_i \exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_i)}$$

If we want $\pi(a|\boldsymbol{x})$ to be restricted to producing a MIPS

Single Item Recommendation (Decision Rule Optimization)

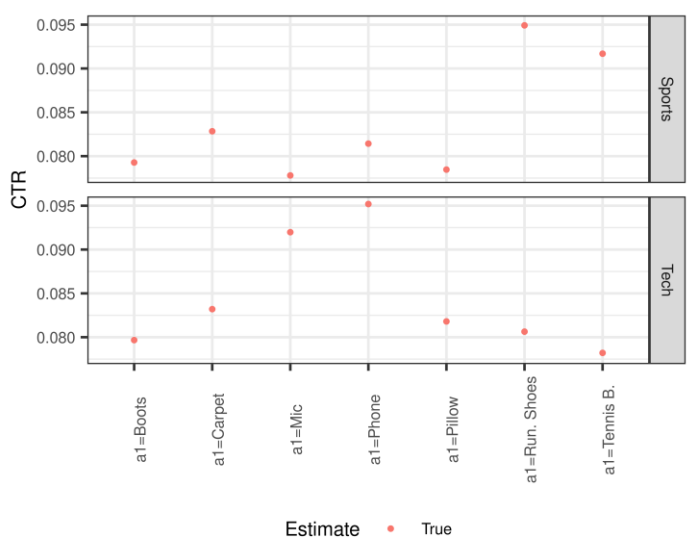
Optimizing the decision rule via a policy

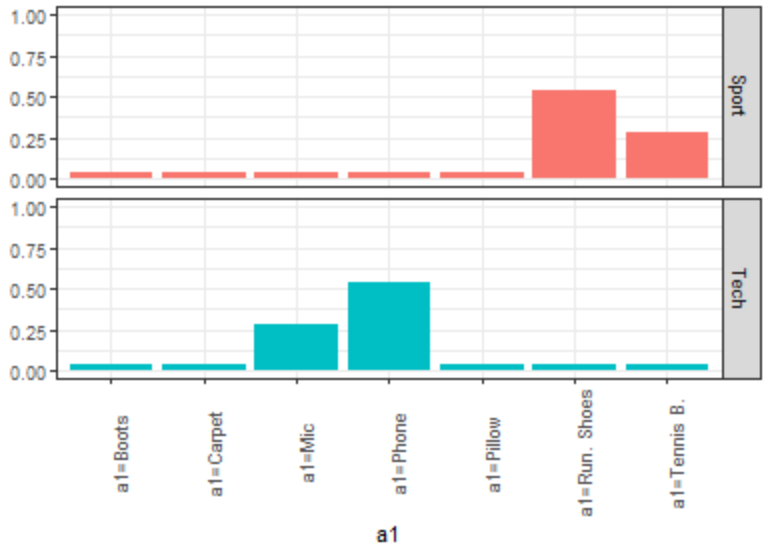




Single Item Recommendation (Decision Rule Optimization)

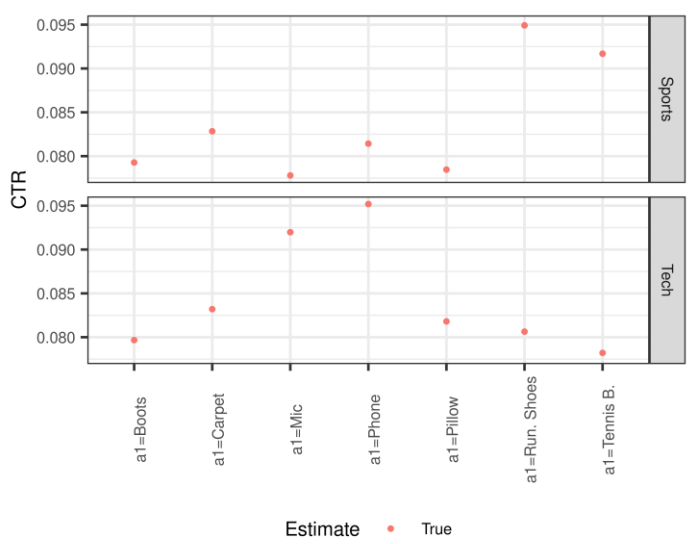
Optimizing the decision rule via a policy

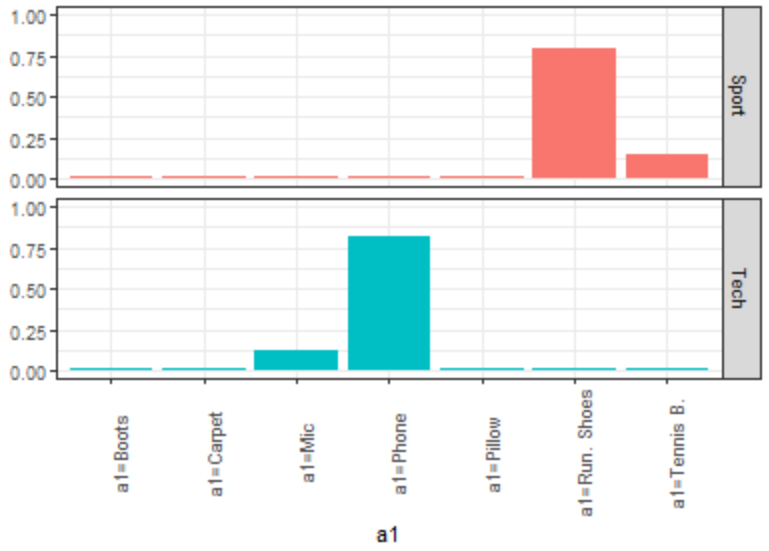




Single Item Recommendation (Decision Rule Optimization)

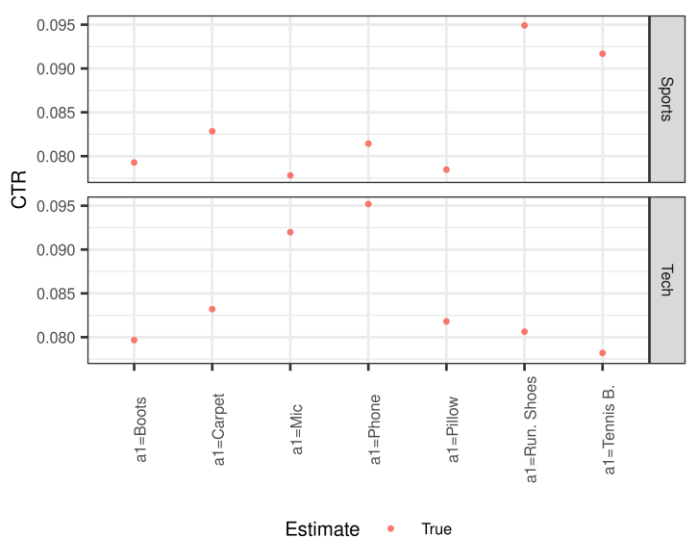
Optimizing the decision rule via a policy

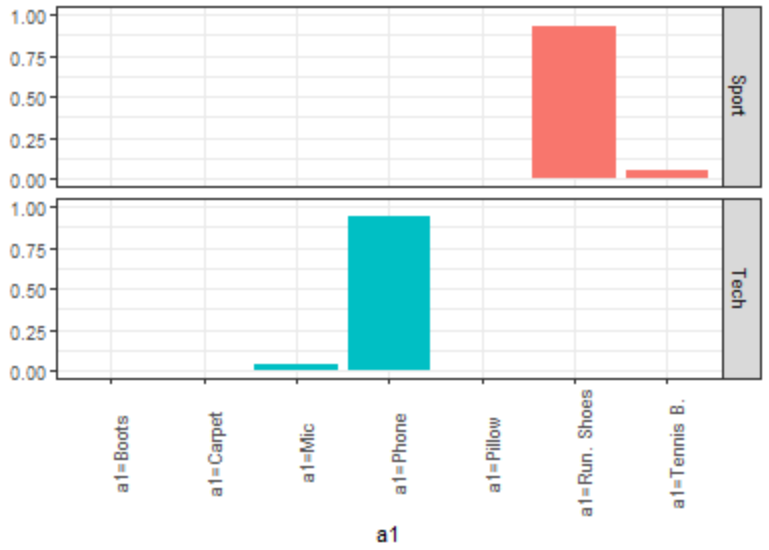




Single Item Recommendation (Decision Rule Optimization)

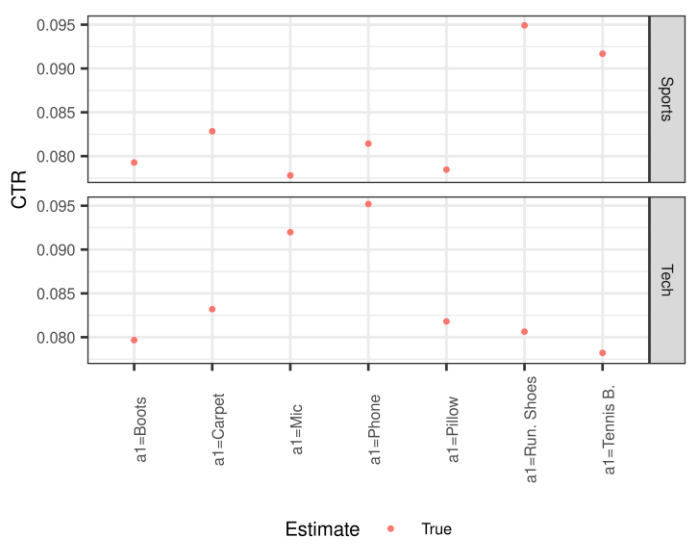
Optimizing the decision rule via a policy

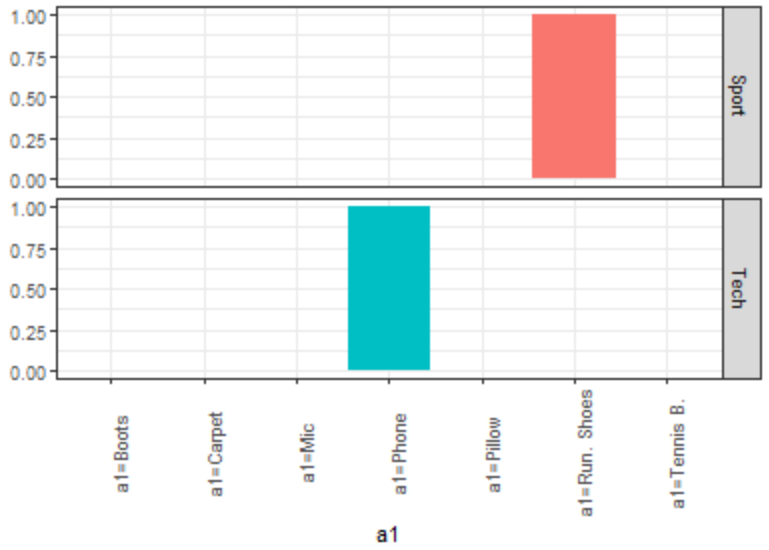




Single Item Recommendation (Decision Rule Optimization)

Optimizing the decision rule via a policy



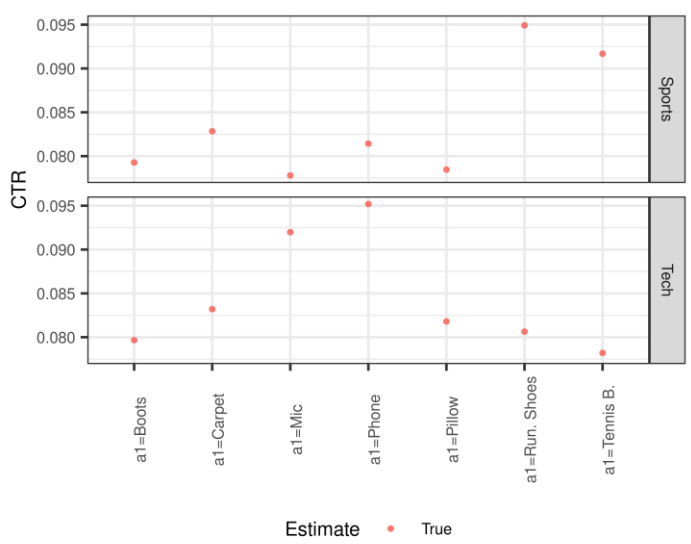


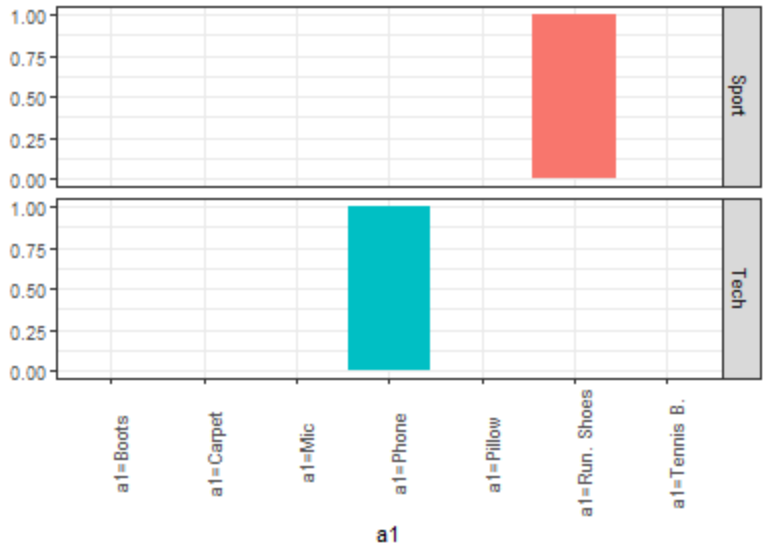
At convergence $\pi(a|\boldsymbol{x})$ is deterministic.

CRITEO

Single Item Recommendation (Decision Rule Optimization)

Optimizing the decision rule via a policy





At convergence $\pi(a|\boldsymbol{x})$ is deterministic.

If the reward estimate is wrong, it will (possibly) select the wrong best action.

CRITEO

Single Item Recommendation (Decision Rule Optimization)

Optimizing the MIPS with Policy Learning

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(a,\boldsymbol{x})\right] \text{ where } \pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_a)}{\sum_i \exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_i)}$$

Exact Gradients:

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \nabla_{\Xi,\beta}\mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}[\hat{R}(a,\boldsymbol{x})] = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(a,\boldsymbol{x})\,\nabla_{\Xi,\beta}\log\pi(a|\boldsymbol{x})\right]$$

Swaminathan, Adith, and Thorsten Joachims. 2015.

Joachims, Thorsten, Adith Swaminathan, and Maarten De Rijke. 2018.

Single Item Recommendation (Decision Rule Optimization)

Optimizing the MIPS with REINFORCE

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(a,\boldsymbol{x})\right] \text{ where } \pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_a)}{\sum_i \exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_i)}$$

Approximate Gradients:

$$a' \sim \pi_{\Xi,\beta}(\cdot\,|\boldsymbol{x})$$

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \hat{R}(\boldsymbol{x},a')\,\nabla_{\Xi,\beta}\log\pi(a'|\boldsymbol{x})$$

On its own we don't get a speed up for using this algorithm.

Chen, et al. 2018.

Single Item Recommendation

SNIPS REINFORCE

CRITEO

Single Item Recommendation (SNIPS REINFORCE)

Optimizing the Objective

Our objective is:

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \nabla_{\Xi,\beta}\mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(\boldsymbol{x},a)\right] = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(\boldsymbol{x},a)\,\nabla_{\Xi,\beta}\log\pi_{\Xi,\beta}(a|\boldsymbol{x})\right]$$

There are numerous problems with this gradient :

- $\log\pi_{\Xi,\beta}(a|x)$ involves the computation of the normalizing constant ➔ O(P)
- If we want to compute the exact expectation ➔ O(P) (Infeasible)
- If we want to approximate the gradient by sampling ➔ O(P)

Everything scales **at least linearly in P** which can be very costly in large catalog sizes.

Single Item Recommendation (SNIPS REINFORCE)

Optimizing the Objective is not Maximum Likelihood

$$\log P(a|\boldsymbol{x}, \boldsymbol{\beta}, \Xi) = f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_a - \log \sum_{a'} \exp(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})$$

$$\nabla_{\beta,\Xi} \log P(a|\boldsymbol{x}, \boldsymbol{\beta}, \Xi) = \nabla_{\beta,\Xi} f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_a - \mathbb{E}_{P(a'|\boldsymbol{x}, \boldsymbol{\beta}, \Xi)}[\nabla_{\beta,\Xi} f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_{a'}]$$

Lots of work on how to solve this problem! e.g.,

Bengio, Yoshua, and Jean-Sébastien Senécal. 2003.

Ruiz, Francisco, et al. 2018.

Rawat, Ankit Singh, et al. 2019.

Single Item Recommendation (SNIPS REINFORCE)

Optimizing the Objective

Our objective is:

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(a,\boldsymbol{x})\right] \quad \text{where } \pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_a)}{\sum_i \exp(f_{\Xi}(\boldsymbol{x})^T\boldsymbol{\beta}_i)}$$

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \mathbb{E}_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}\left[\hat{R}(\boldsymbol{x},a)\,\nabla_{\Xi,\beta}\log\pi_{\Xi,\beta}(a|\boldsymbol{x})\right]$$

→ We propose a new gradient :

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = Cov_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}[\hat{R}(\boldsymbol{x},a), \nabla_{\Xi,\beta}f_{\Xi}(\boldsymbol{x})^{\boldsymbol{T}}\boldsymbol{\beta}_a]$$

$$\text{with } Cov[Y,\ \boldsymbol{Z}] = \mathbb{E}[(Y - \mathbb{E}[Y])(\boldsymbol{Z} - \mathbb{E}[\boldsymbol{Z}])]$$

Single Item Recommendation (SNIPS REINFORCE)

Self Normalized Importance Sampling

- Our new gradient gets rid of the normalizing constant, is that sufficient?

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = Cov_{a\sim\pi_{\Xi,\beta}(.|\boldsymbol{x})}[\hat{R}(\boldsymbol{x},a), \nabla_{\Xi,\beta} f_{\Xi}(\boldsymbol{x})^{T}\boldsymbol{\beta}_a]$$

NO! We still need to avoid sampling from $\pi_{\Xi,\beta}(.|\boldsymbol{x})$ to approximate the gradient.

Solution : Self Normalized Importance Sampling!

Single Item Recommendation (SNIPS REINFORCE)

Self Normalized Importance Sampling

If we have a proposal Q that is fast to sample from, we can approximate the covariance gradient $Cov[Y,\ \boldsymbol{Z}]$ by $\sum_{i=1}^{S} w_i(Y_i\ - \bar{Y})(Z_i\ - \bar{Z})$ with samples $a_1, \dots, a_S$ from Q with :

$$\tilde{\pi}(a|\boldsymbol{x}) = \exp\left(f_{\Xi}(\boldsymbol{x})^{\boldsymbol{T}}\boldsymbol{\beta}_a\right), \qquad Y_i = Y(a_i), \qquad Z_i = Z(a_i)$$

$$\widetilde{w}_i = \frac{\tilde{\pi}(a_i|\boldsymbol{x})}{Q(a_i|\boldsymbol{x})}, \qquad w_i = \frac{\widetilde{w}_i}{\sum \widetilde{w}_j}, \qquad \bar{Y} = \sum_{i=1}^{N} w_i Y_i\,, \qquad \bar{Z} = \sum_{i=1}^{N} w_i Z_i$$

→ How do we choose Q?

Single Item Recommendation (SNIPS REINFORCE)

Proposal Selection

We need an additional Assumption :

$$\pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})}{\sum_{a'} \exp(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})}$$

Assume $\boldsymbol{\beta}$ is ***fixed*** and we only optimize $f_{\Xi}(\boldsymbol{x})$. This structure allows fast search in *O(log P).*

MIPS give us $\alpha_K(x) = \text{argsort}(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})_{1:K}$ the top-K (approximate) items of the policy

We define :

$$Q_{\epsilon,K}(a|\boldsymbol{x}) = \begin{cases} \frac{\epsilon}{P} + (1 - \epsilon)\kappa(a|\boldsymbol{x}) \; if \; a \in \alpha_K(\boldsymbol{x}) \\ \frac{\epsilon}{P} \; else \end{cases} \quad \text{with } \kappa(a|\boldsymbol{x}) = \frac{\exp(f_{\Xi}(\boldsymbol{x})^{\boldsymbol{T}} \boldsymbol{\beta}_a)}{\sum_{a' \in \alpha_K(x)} \exp(f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})}$$

Single Item Recommendation (SNIPS REINFORCE)

Experiments

We test the method on a session completion task : X observed session, Y complementary items.

This can be cast into an offline bandit framework where our policy π_θ takes the observed part X as a context, recommends an item a and receives the binary reward $\hat{r}(a, X) = 1[a \in Y]$.

We use X to build item embeddings $\boldsymbol{\beta}$ of dimension $L \ll P$.

Define x_{emb} as the item embeddings average in session X.

$$\pi(a|x) = \frac{\exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a)}{\sum_{a'} \exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})}$$

	Catalog size	Number of users
Twitch	750K	500K
GoodReads	1.23M	300K

Single Item Recommendation (SNIPS REINFORCE)

Experiments

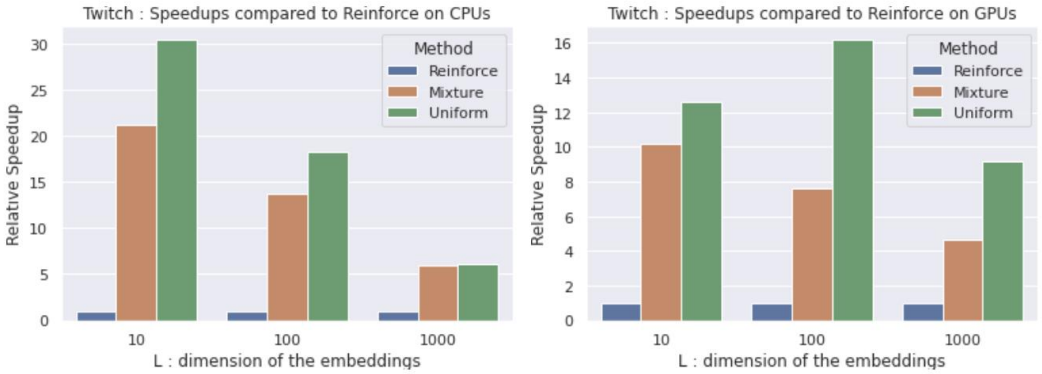
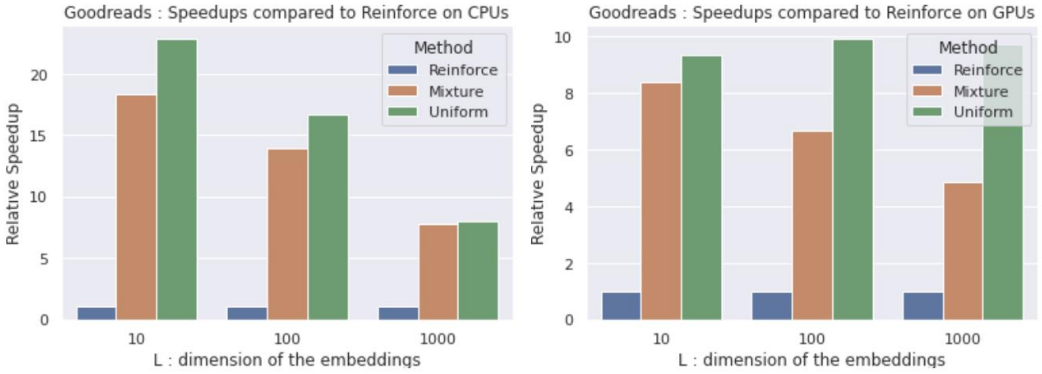


Fig. 1. The Twitch Dataset : The speedup of the proposed algorithms on both CPU and GPU devices



CRITEO

Single Item Recommendation (SNIPS REINFORCE)

Experiments

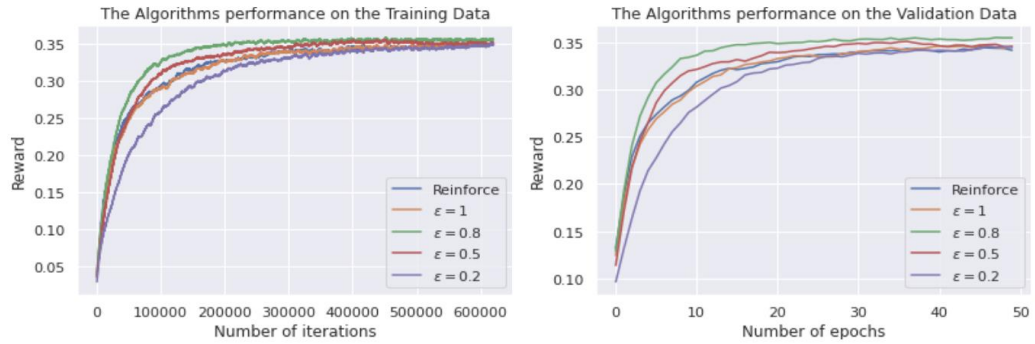


Fig. 3. The Twitch Dataset : ϵ effect on the policy training and validation

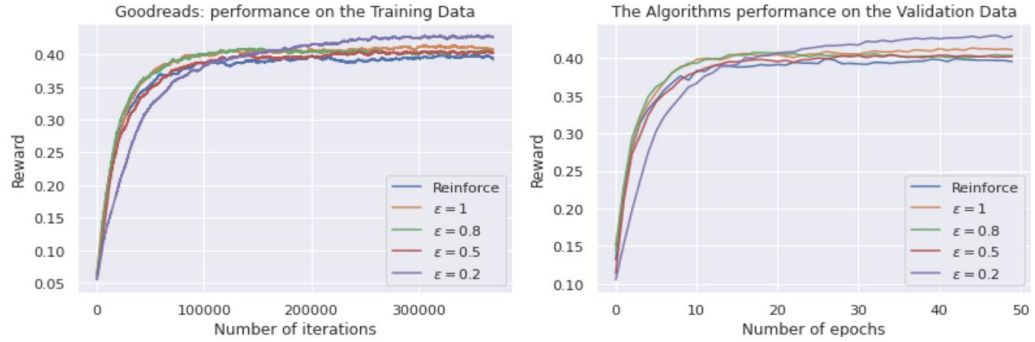


Fig. 4. The Goodreads Dataset : ϵ effect on the policy training and validation

CRITEO

Single Item Recommendation (SNIPS REINFORCE)

Experiments

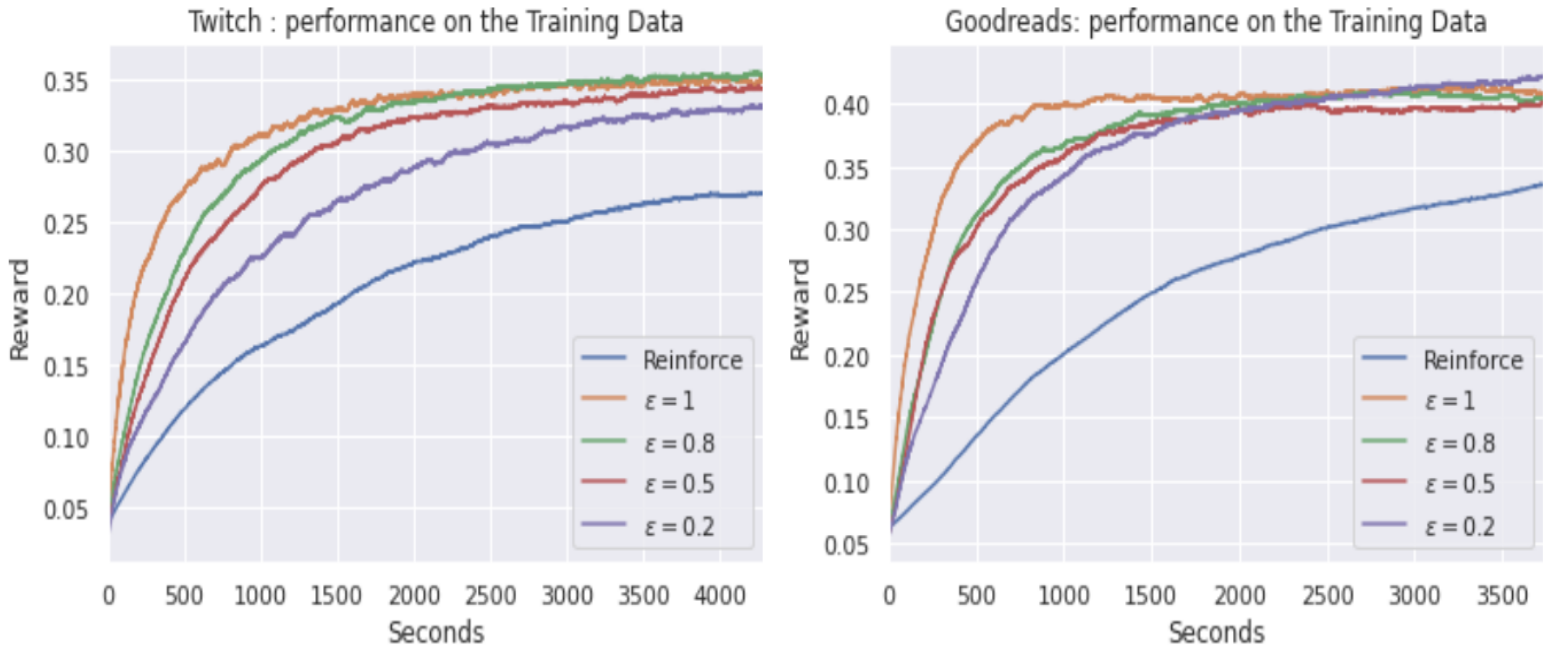


Fig. 5. Performance plots given time : ϵ's effect on the training

[Sakhi et al. 2022]

SNIPS Reinforce (Practice)

Python Tutorial 2

https://colab.research.google.com/github/otmhi/Reward-Optimizing-Reco/blob/main/Reward_Optimizing_Slate_Recommendation_with_DL_and_MIPS_Part_2.ipynb

Slate Recommendation

Motivation

CRITEO

Slate Recommendation (Motivation)

Recommender Systems in Real-World

product viewed

CRITEO

Slate Recommendation (Motivation)

Recommender Systems in Real-World

product viewed

product viewed

CRITEO

Slate Recommendation (Motivation)

Recommender Systems in Real-World

product viewed

product viewed

recommended slate:

CRITEO

Slate Recommendation (Motivation)

Recommender Systems in Real-World

product viewed

product viewed

recommended slate:

No click No click click

CRITEO

Slate Recommendation Framework

CRITEO

Slate Recommendation (Framework)

From Single Items to Slates

- The recommender system observes a context $\boldsymbol{x} \sim \nu$.
- The recommender system selects an ordered list (or slate) of K items $a_1, a_2, \dots, a_K$.
- The recommender system receives a reward $R \sim P_R(\cdot | \boldsymbol{x}, a_1, \dots, a_K)$ and a list of ranks $r_1, r_2, \dots, r_K \sim P_r(\cdot | \boldsymbol{x}, s)$.

→ Similarly, to single item recommendation, the recommender system selects a slate of K items $a_1, a_2, \dots, a_K$ following a decision rule $B(\cdot)$ that boils down to a MIPS task.

Slate Recommendation (Framework)

From Single Items to Slates

- The space of possible actions (i.e., slates) is huge compared to single item recommendation (P^K).
- Estimating the reward is more difficult, the reward signal is spread over a huge combination of context-slate pairs.
- Additional biases related to the layout of the slates (e.g., position bias).
- Additional factors that might have a large contribution to the reward like the size of the slate…
- Optimizing the policy is difficult…

Slate Recommendation (Framework)

From Single Items to Slates

- Given a policy π, the decision rule in single item recommendation is:

$$\text{argmax}_a \pi(a \mid \boldsymbol{x})$$

  - We want to reduce this decision rule to a MIPS: $\text{argmax}_a f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a$
  - This is achievable using softmax: $\pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a)}{\sum_{a'} \exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})}$, in which case

$$\text{argmax}_a \pi_{\Xi,\beta}(a \mid \boldsymbol{x}) = \text{argmax}_a f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a$$

- In slate recommendation, the decision rule becomes:

$$\text{argmax}_{a_1,a_2,\ldots,a_K} \pi(a_1, a_2, \ldots, a_K \mid \boldsymbol{x})$$

  - We want to reduce to a MIPS: $(\text{argsort } f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta})_{1:K}$

Slate Recommendation (Framework)

From Single Items to Slates

- Given a policy π, the decision rule in single item recommendation is:

$$\text{argmax}_a \pi(a \mid \boldsymbol{x})$$

  - We want to reduce this decision rule to a MIPS: $\text{argmax}_a f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a$

  - This is achievable using softmax: $\pi_{\Xi,\beta}(a|\boldsymbol{x}) = \frac{\exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a)}{\sum_{a'} \exp(f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_{a'})}$, in which case

$$\text{argmax}_a \pi_{\Xi,\beta}(a \mid \boldsymbol{x}) = \text{argmax}_a f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta}_a$$

- In slate recommendation, the decision rule becomes:

$$\text{argmax}_{a_1,a_2,\ldots,a_K} \pi(a_1, a_2, \ldots, a_K \mid \boldsymbol{x})$$

  - We want to reduce to a MIPS: $(\text{argsort}\ f_\Xi(\boldsymbol{x})^T \boldsymbol{\beta})_{1:K}$

The equivalence between argmax and argsort is not trivial. The softmax parametrization does not solve it!

Slate Recommendation (Framework)

From Single Items to Slates

- In slate recommendation, the decision rule becomes:

$$\text{argmax}_{a_1,a_2,\dots,a_K} \pi(a_1, a_2, \dots, a_K \mid \boldsymbol{x})$$

  - We want to reduce to a MIPS: $\text{argsort}\ (f_{\Xi}(\boldsymbol{x})^T \boldsymbol{\beta})_{1:K}$

The equivalence between argmax and argsort is not trivial. The softmax parametrization does not solve it!

→ Need of a clever parametrization of the models and policies...

Slate Recommendation

IPS Method

CRITEO

Slate Recommendation (IPS Method)

From Single Items to Slates

- Recall that the performance of recommendation policies π is often quantified by the value function:

$$V(\pi) = \mathbb{E}_{x\sim\nu}\mathbb{E}_{a_1,\dots,a_K\sim\pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a_1, \dots, a_K)] = \mathbb{E}_{x\sim\nu}[\sum_{a_1}\dots\sum_{a_K}\pi(a_1, \dots, a_K \mid \boldsymbol{x})R(\boldsymbol{x}, a_1, \dots, a_K)]$$

Slate Recommendation (IPS Method)

From Single Items to Slates

- Recall that the performance of recommendation policies π is often quantified by the value function:

$$V(\pi) = \mathbb{E}_{x\sim\nu}\mathbb{E}_{a_1,\dots,a_K\sim\pi(\cdot|\boldsymbol{x})}[R(\boldsymbol{x}, a_1, \dots, a_K)] = \mathbb{E}_{x\sim\nu}[\sum_{a_1}\dots\sum_{a_K}\pi(a_1, \dots, a_K \mid \boldsymbol{x})R(\boldsymbol{x}, a_1, \dots, a_K)]$$

- The corresponding IPS estimator is

$$\hat{V}_n^{IPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n} R_i \frac{\pi(a_{i,1}, \dots, a_{i,K} \mid \boldsymbol{x}_i)}{\pi_0(a_{i,1}, \dots, a_{i,K} \mid \boldsymbol{x}_i)}$$

Slate Recommendation (IPS Method)

From Single Items to Slates

- The IPS estimator of the value function is

$$\hat{V}_n^{IPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n} R_i \frac{\pi(a_{i,1}, \dots, a_{i,K} \mid \boldsymbol{x}_i)}{\pi_0(a_{i,1}, \dots, a_{i,K} \mid \boldsymbol{x}_i)}$$

- It is unbiased.
- However, it's variance scales linearly with $\frac{\pi(a_{i,1},\dots,a_{i,K}|\boldsymbol{x}_i)}{\pi_0(a_{i,1},\dots,a_{i,K}|\boldsymbol{x}_i)}$ which can be huge.
- **Example:** The logging policy π_0 is uniform. In this case, the variance grows linearly with $P^K\pi(a|x)$, where P is the catalog size.

Slate Recommendation (IPS Method)

From Single Items to Slates

Solution: Independent IPS (IIPS)

- (A1) The expected reward of a slate is the sum of expected ranks of its items:

$$R(\boldsymbol{x}, a_1, \dots, a_K) = \sum_{k \in [K]} r_k(\boldsymbol{x}, a_1, \dots, a_K)$$

- (A2) The expected rank only depends on the item and its position:

$$r_k(\boldsymbol{x}, a_1, \dots, a_K) = r_k(\boldsymbol{x}, a_k)$$

- The Independent IPS (IIPS) estimator of the value function follows as

$$\hat{V}_n^{IIPS}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} r_{i,k} \frac{\pi(a_{i,k}, k \mid \boldsymbol{x}_i)}{\pi_0(a_{i,k}, k \mid \boldsymbol{x}_i)}$$

Slate Recommendation (IPS Method)

From Single Items to Slates

Solution: Independent IPS (IIPS)

- (A1) The expected reward of a slate is the sum of expected ranks of its items:

$$R(\boldsymbol{x}, a_1, \dots, a_K) = \sum_{k \in [K]} r_k(\boldsymbol{x}, a_1, \dots, a_K)$$

- (A2) The expected rank only depends on the item and its position:

$$r_k(\boldsymbol{x}, a_1, \dots, a_K) = r_k(\boldsymbol{x}, a_k)$$

- The Independent IPS (IIPS) estimator of the value function follows as

$$\hat{V}_n^{IIPS}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} r_{i,k} \frac{\pi(a_{i,k}, k \mid \boldsymbol{x}_i)}{\pi_0(a_{i,k}, k \mid \boldsymbol{x}_i)}$$

Marginal probability that policy π places item $a_{i,k}$ in position k

Slate Recommendation (IPS Method)

From Single Items to Slates

Solution: Independent IPS (IIPS)

- (A1) The expected reward of a slate is the sum of expected ranks of its items:

$$R(\boldsymbol{x}, a_1, \dots, a_K) = \sum_{k \in [K]} r_k(\boldsymbol{x}, a_1, \dots, a_K)$$

- (A2) The expected rank only depends on the item and its position:

$$r_k(\boldsymbol{x}, a_1, \dots, a_K) = r_k(\boldsymbol{x}, a_k)$$

- The Independent IPS (IIPS) estimator of the value function follows as

$$\hat{V}_n^{IIPS}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} r_{i,k} \frac{\pi(a_{i,k}, k \mid \boldsymbol{x}_i)}{\pi_0(a_{i,k}, k \mid \boldsymbol{x}_i)}$$

← Hard to compute

Slate Recommendation (IPS Method)

From Single Items to Slates

- The Independent IPS (IIPS) estimator of the value function is

$$\hat{V}_n^{IIPS}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} r_{i,k}\frac{\pi(a_{i,k}, k \mid \boldsymbol{x}_i)}{\pi_0(a_{i,k}, k \mid \boldsymbol{x}_i)}$$

- The IIPS estimator is unbiased under (A1) and (A2).
- The variance is reduced compared to IPS as it scales linearly with $\frac{\pi(a_{i,k},k|\boldsymbol{x}_i)}{\pi_0(a_{i,k},k|\boldsymbol{x}_i)}$ instead of $\frac{\pi(a_{i,1},\ldots,a_{i,K}|\boldsymbol{x}_i)}{\pi_0(a_{i,1},\ldots,a_{i,K}|\boldsymbol{x}_i)}$. **Example:** In the case of uniform logging policies, the variance is reduced from $P^K\pi(a|x)$ to $P\pi(a|x)$.

Slate Recommendation (IPS Method)

From Single Items to Slates

- IIPS is unbiased and reduces the variance of IPS. However, IIPS makes the strong independence assumption.
- Also, IIPS requires computing the marginal distribution $\pi(a_{i,k}, k \mid \boldsymbol{x}_i)$, which is often intractable since

$$\pi(a', k \mid \boldsymbol{x}_i) = \underbrace{\sum_{a_1} \dots \sum_{a_K}}_{\text{This is totally infeasible}} \pi(a_1, .., a_K | \boldsymbol{x}) \mathbb{I}_{a_k = a'}$$

Slate Recommendation (IPS Method)

From Single Items to Slates

- IIPS is unbiased and reduces the variance of IPS. However, IIPS makes the strong independence assumption.
- Also, IIPS requires computing the marginal distribution $\pi(a_{i,k}, k \mid \boldsymbol{x}_i)$, which is often intractable since

$$\pi(a', k \mid \boldsymbol{x}_i) = \sum_{a_1} \dots \sum_{a_K} \pi(a_1, .., a_K | \boldsymbol{x}) \mathbb{I}_{a_k = a'}$$

Common Solution: Factorized policies

$$\pi(a_1, .., a_K | \boldsymbol{x}) = \prod_{k=1}^{K} \pi(a_k, k \mid \boldsymbol{x})$$

Slate Recommendation (IPS Method)

From Single Items to Slates

- IIPS is unbiased and reduces the variance of IPS. However, IIPS makes the strong independence assumption.
- Also, IIPS requires computing the marginal distribution $\pi(a_{i,k}, k \mid \boldsymbol{x}_i)$, which is often intractable since

$$\pi(a', k \mid \boldsymbol{x}_i) = \sum_{a_1} \dots \sum_{a_K} \pi(a_1, .., a_K | \boldsymbol{x}) \mathbb{I}_{a_k = a'}$$

Common Solution: Factorized policies

$$\pi(a_1, .., a_K | \boldsymbol{x}) = \prod_{k=1}^{K} \pi(a_k, k \mid \boldsymbol{x})$$

This is the marginal probability that policy π places item a_k in position k

CRITEO

Slate Recommendation (IPS Method)

From Single Items to Slates

- IIPS is unbiased and reduces the variance of IPS. However, IIPS makes the strong independence assumption.
- Also, IIPS requires computing the marginal distribution $\pi(a_{i,k}, k \mid \boldsymbol{x}_i)$, which is often intractable since $\pi(a', k \mid \boldsymbol{x}_i) = \sum_{a_1} \dots \sum_{a_K} \pi(a_1, .., a_K | \boldsymbol{x}) \mathbb{I}_{a_k = a'}$

Common Solution: Factorized policies

$$\pi_0(a_1, .., a_K | \boldsymbol{x}) = \prod_{k=1}^{K} \pi_0(a_k, k \mid \boldsymbol{x})$$

$$\pi(a_1, .., a_K | \boldsymbol{x}) = \prod_{k=1}^{K} \pi(a_k, k \mid \boldsymbol{x})$$

Problems:

- The logging policy $\pi_0$is not necessarily factorized.
- Best slate by π contains the best item repeated K times.

Slate Recommendation

Direct Method

CRITEO

Slate Recommendation (Direct Method)

Rank vs. Reward

- Ranking Model (e.g., BPR):
  - Trained on clicked slates only.
  - Not reward optimizing, it does not see it.
- Reward Model:
  - Trained on all slates.
  - Returns calibrated CTRs.

Slate Recommendation (Direct Method)

Rank vs. Reward

- Ranking Model (e.g., BPR):
  - Trained on clicked slates only.
  - Not reward optimizing, it does not see it.

- Reward Model:
  - Trained on all slates.
  - Returns calibrated CTRs.

WE CAN COMBINE BOTH → Probabilistic Reward and Rank (PRR (Aouali et al. 2022))

Slate Recommendation (Direct Method)

PRR Model (Aouali et al. 2022)

The Probabilistic Rank and Reward Model (PRR) has the following properties:

- We assume at most one element in the slate is clicked.
  - This differs with IIPS, Position Based Model, etc.
- PRR includes user engagement features and recommendation features.
- PRR incorporates:
  - was the slate clicked? bandit approach
  - which item was clicked? IIPS, position-based models
- Decision making in PRR boils down to a MIPS task.

Details in: Aouali et al. 2022

Slate Recommendation (Direct Method)

PRR Model (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

$$\theta_0 = \exp(\boldsymbol{y}^T \boldsymbol{\phi})$$

$\boldsymbol{y}$ – User engagement features (slate level)
$\boldsymbol{\phi}$ – Nuisance parameters

Slate Recommendation (Direct Method)

PRR Model (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \underbrace{\frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}}\right), \qquad Z = \sum \theta_k$$

$$\theta_k = \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k) + \exp(\alpha_k)$$

$\mathbf{z}$ – context
$g_\Gamma(\mathbf{z})$ – contextual embedding
Ψ_a – Item embedding for item a
γ, α – multiplicative and additional position bias

Slate Recommendation (Direct Method)

PRR Learning (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \underbrace{\frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}}\right), \qquad Z = \sum \theta_k$$

$$\theta_K = \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k) + \exp(\alpha_k)$$

$\mathbf{z}$ – context
$g_\Gamma(\mathbf{z})$ – contextual embedding
Ψ_a – Item embedding for item a
γ, α – multiplicative and additional position bias

→ We use the maximum likelihood principle to estimate the parameters $\mathbf{\Psi}, \mathbf{\Gamma}, \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\phi}$.

Slate Recommendation (Direct Method)

PRR Decision Rule (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

$$\rightarrow P(R = 1 \mid y, z, a_1, \dots, a_K) = 1 - \frac{\theta_0}{Z}$$

The decision rule follows as:

$$\text{argmax}_{a_1,\dots,a_K} P(R = 1 \mid y, z, a_1, \dots, a_K) = \text{argmax}_{a_1,\dots,a_K} 1 - \frac{\theta_0}{Z} = \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{Z}$$

Slate Recommendation (Direct Method)

PRR Decision Rule (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

$$\rightarrow P(R = 1 \mid y, z, a_1, \dots, a_K) = 1 - \frac{\theta_0}{Z}$$

The decision rule follows as:

$$\text{argmax}_{a_1,\dots,a_K} P(R = 1 \mid y, z, a_1, \dots, a_K) = \text{argmax}_{a_1,\dots,a_K} 1 - \frac{\theta_0}{Z} = \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{Z}$$

$$= \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{\theta_0 + \sum \theta_k} = \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \theta_k$$

Slate Recommendation (Direct Method)

PRR Decision Rule (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

$$\rightarrow P(R = 1 \mid y, z, a_1, \dots, a_K) = 1 - \frac{\theta_0}{Z}$$

The decision rule follows as:

$$\text{argmax}_{a_1,\dots,a_K} P(R = 1 \mid y, z, a_1, \dots, a_K) = \text{argmax}_{a_1,\dots,a_K} 1 - \frac{\theta_0}{Z} = \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{Z}$$

$$= \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{\theta_0 + \sum \theta_k} = \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \theta_k$$

$$= \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k) + \exp(\alpha_k)$$

Slate Recommendation (Direct Method)

PRR Decision Rule (Aouali et al. 2022)

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

$$\rightarrow P(R = 1 \mid y, z, a_1, \dots, a_K) = 1 - \frac{\theta_0}{Z}$$

The decision rule follows as:

$$\text{argmax}_{a_1,\dots,a_K} P(R = 1 \mid y, z, a_1, \dots, a_K) = \text{argmax}_{a_1,\dots,a_K} 1 - \frac{\theta_0}{Z} = \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{Z}$$

$$= \text{argmin}_{a_1,\dots,a_K} \frac{\theta_0}{\theta_0 + \sum \theta_k} = \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \theta_k$$

$$= \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k) + \exp(\alpha_k)$$

$$= \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k)$$

Slate Recommendation (Direct Method)

PRR Decision Rule (Aouali et al. 2022)

The decision rule follows as:

$$\text{argmax}_{a_1,\dots,a_K} P(R = 1 \mid y, z, a_1, \dots, a_K) = \text{argmax}_{a_1,\dots,a_K} \sum_{k=1}^{K} \exp(g_\Gamma(\mathbf{z})^T \mathbf{\Psi}_{a_k}) \exp(\gamma_k)$$

The argmax operation on the right-hand-side can be solved using MIPS as follows:

$$a'_1, \dots, a'_K = \text{argsort}(g_\Gamma(\boldsymbol{z})^T \mathbf{\Psi})_{1:K}$$

$$i_1, \dots, i_K = \text{argsort}(\boldsymbol{\gamma})$$

$$a_1, \dots, a_K = a'_{i_1}, \dots, a'_{i_K}$$

Slate Recommendation (Direct Method)

PRR Variants (Aouali et al. 2022)

PRR model

$$\bar{R}, r_1, \dots, r_K | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum \theta_k$$

PRR Reward Only

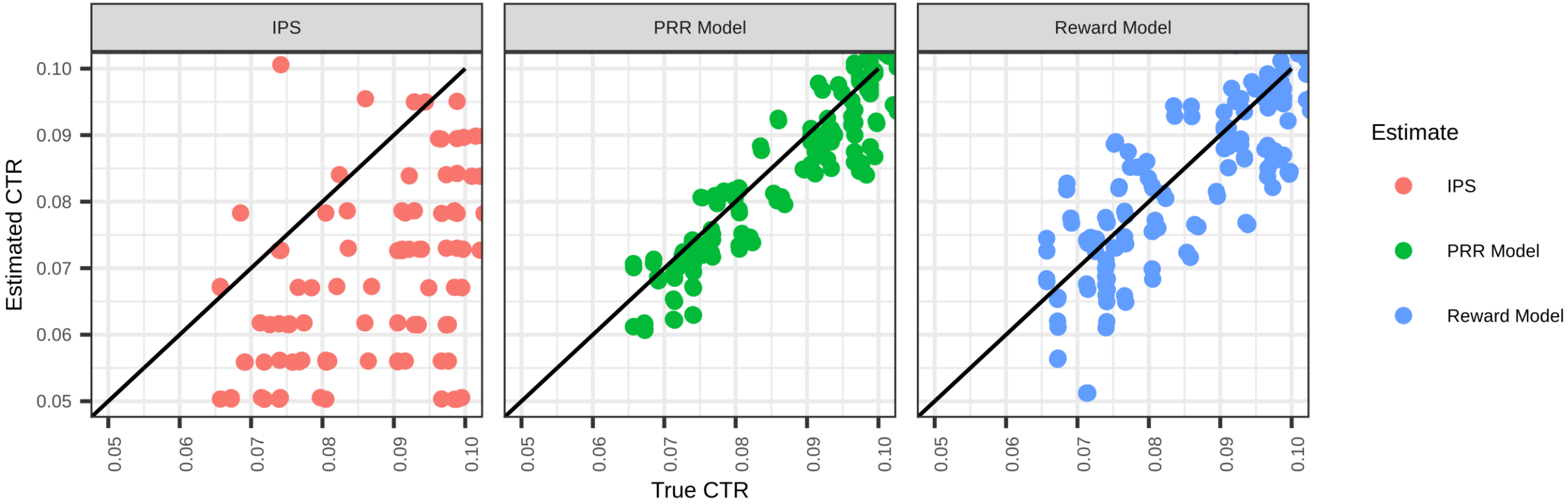
$$\bar{R}, R | \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_0}{Z}, \frac{\sum \theta_k}{Z}\right), \qquad Z = \sum_{k=0}^{K} \theta_k$$

PRR Rank Only

$$r_1, \dots, r_K | R = 1, \mathbf{y}, \mathbf{z}, a_1, \dots, a_K \sim \text{categorical}\left(\frac{\theta_1}{Z}, \dots, \frac{\theta_K}{Z}\right), \qquad Z = \sum_{k=1}^{K} \theta_k$$

$$\theta_0 = \exp(\boldsymbol{y}^T \boldsymbol{\phi}) \text{ and } \theta_k = \exp(g_\Gamma(\mathbf{z})^T \boldsymbol{\Psi}_{ak}) \exp(\gamma_k) + \exp(\alpha_k)$$

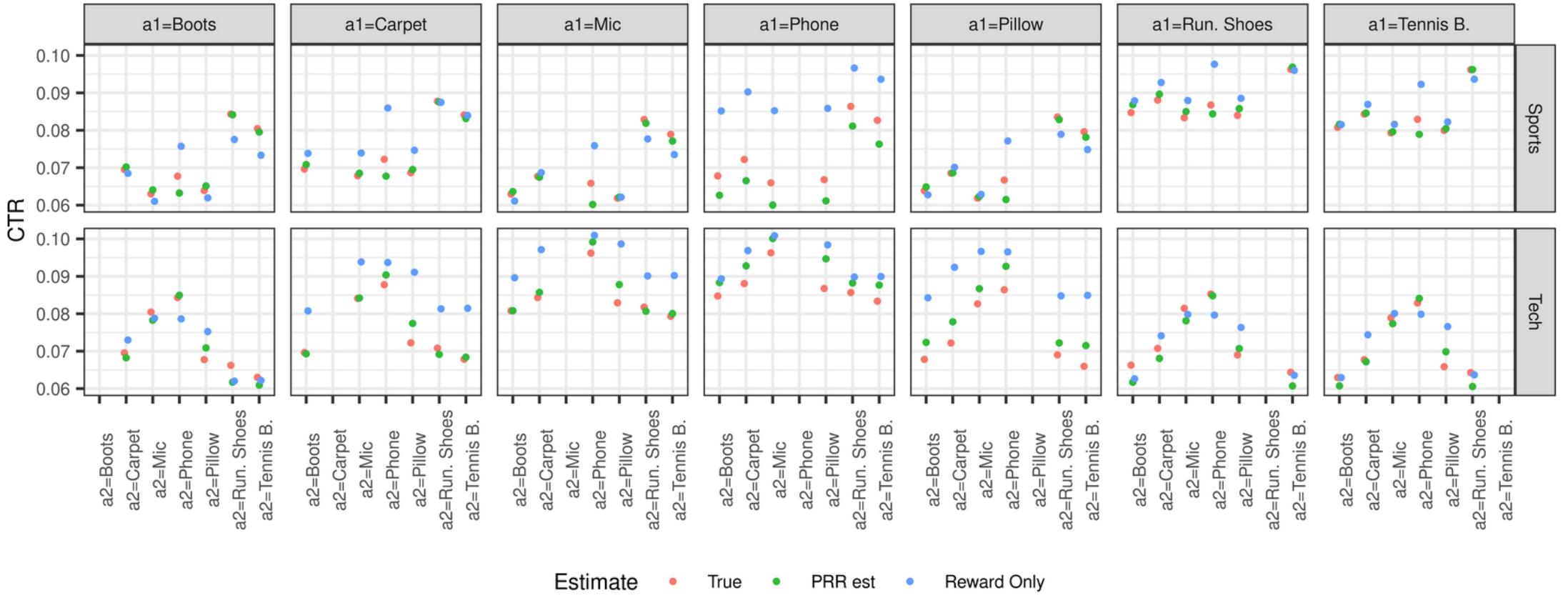
Slate Recommendation (Direct Method)



CRITEO

Slate Recommendation (Direct Method)

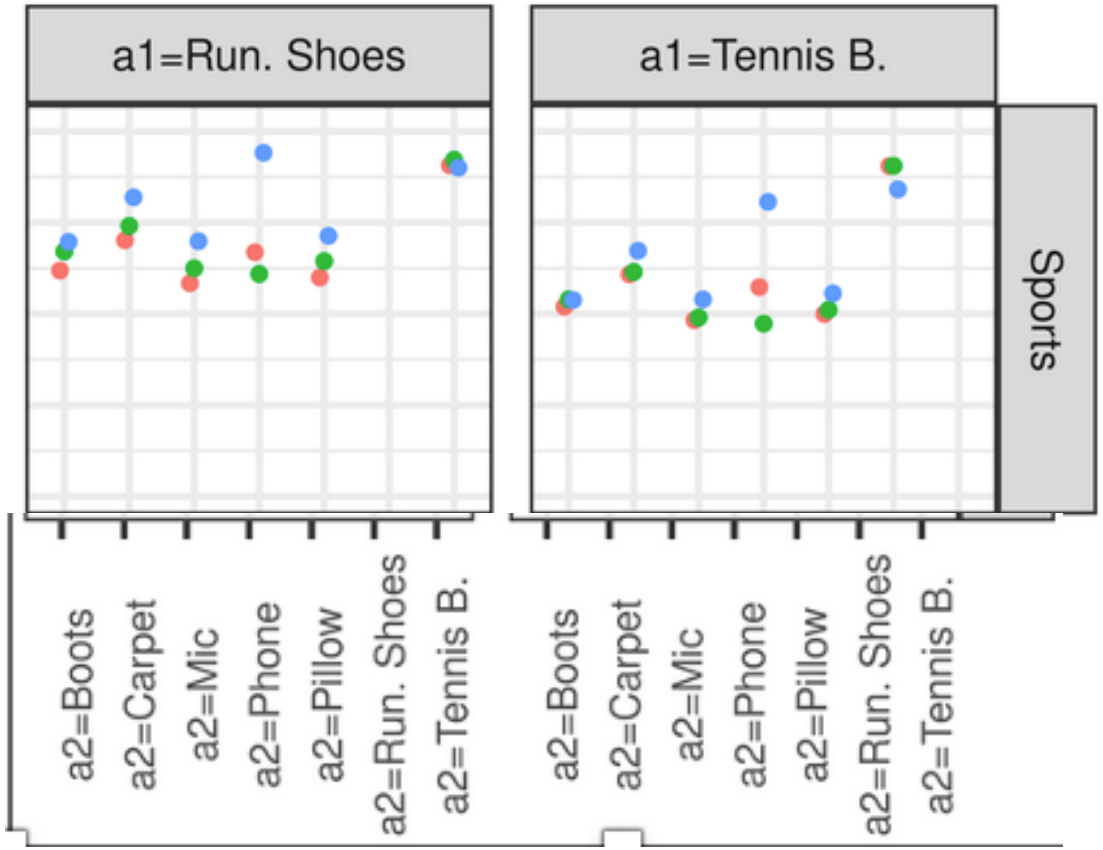
PRR Toy Example: CTR Estimates



CRITEO

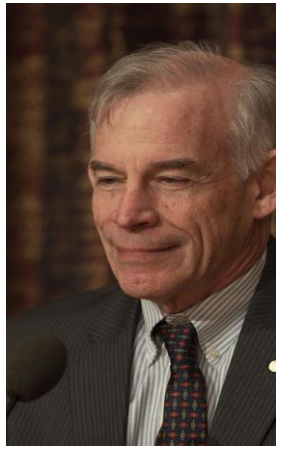
Slate Recommendation (Direct Method)

PRR Toy Example: CTR Estimates



CRITEO

Slate Recommendation (DM vs. IPS)



- Christopher Sims and Larry Wasserman had an epic debate on the IPS (or Horvitz-Thompson) estimation: Whether it causes a problem for likelihood/Bayesian inference.
- Bayesians/Likelihoodists should not use the propensity score…
- PRR + BLOB ignore the propensity score and outperform other methods.

Debate and discussion:

Bayesian Causal Inference for Real World Interactive Systems (bcirwis2021.github.io)

Christopher Sims - Large Parameter Spaces and Weighted Data: A Bayesian Perspective - YouTube

Slate Recommendation

Decision Rule Optimization

CRITEO

Slate Recommendation (Decision Rule Optimization)

Policy Learning for Slates

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \sum_{a_1} \dots \sum_{a_K} \hat{R}(x, a_1, .., a_K)\pi_{\Xi,\beta}(a_1, .., a_K|\boldsymbol{x})$$

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \nabla_{\Xi,\beta} \underbrace{\sum_{a_1} \dots \sum_{a_K}} \hat{R}(x, a_1, .., a_K)\pi_{\Xi,\beta}(a_1, .., a_K|\boldsymbol{x})$$

This is now totally infeasible

Slate Recommendation (Decision Rule Optimization)

Reinforce Algorithm for Slates

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \sum_{a_1} \dots \sum_{a_K} \hat{R}(x, a_1, .., a_K)\pi_{\Xi,\beta}(a_1, .., a_K|\boldsymbol{x})$$

Solution:

$$a'_1, \dots, a'_K \sim \pi_{\Xi,\beta}(\cdot\,|\boldsymbol{x})$$

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \hat{R}(x, a'_1, \dots, a'_K)\nabla_{\Xi,\beta}\log \pi_{\Xi,\beta}(a'_1, \dots, a'_K|\boldsymbol{x})$$

If we can sample and evaluate $\pi_{\Xi,\beta}(\cdot\,|\boldsymbol{x})$ then we can proceed…

Slate Recommendation (Decision Rule Optimization)

Reinforce Algorithm for Slates

$$\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \sum_{a_1} \dots \sum_{a_K} \hat{R}(x, a_1, .., a_K)\pi_{\Xi,\beta}(a_1, .., a_K|\boldsymbol{x})$$

Solution:

$$a'_1, \dots, a'_K \sim \pi_{\Xi,\beta}(\cdot\,|\boldsymbol{x})$$

$$\nabla_{\Xi,\beta}\hat{V}_{\Xi,\beta}(\boldsymbol{x}) = \hat{R}(x, a'_1, \dots, a'_K)\nabla_{\Xi,\beta}\log \pi_{\Xi,\beta}(a'_1, \dots, a'_K|\boldsymbol{x})$$

If we can sample and evaluate $\pi_{\Xi,\beta}(\cdot\,|\boldsymbol{x})$ then we can proceed...

↑

Factorized policies

Slate Recommendation (Decision Rule Optimization)

Back to Factorized Policies

$$\pi(a_1, \dots, a_K|\boldsymbol{x}) = \prod_k \pi(a_k, k|\boldsymbol{x})$$

What if the best two recommendations are rice and beer. Suppose that

$$\pi(a = \text{rice}|\boldsymbol{x}) = 0.51$$
$$\pi(a = \text{beer}|\boldsymbol{x}) = 0.49$$

Then

$$\pi(a_1 = \text{rice}, a_2 = \text{rice}|\boldsymbol{x}) = 0.2601$$
$$\pi(a_1 = \text{rice}, a_2 = \text{beer}|\boldsymbol{x}) = 0.2499$$
$$\pi(a_1 = \text{beer}, a_2 = \text{rice}|\boldsymbol{x}) = 0.2499$$
$$\pi(a_1 = \text{beer}, a_2 = \text{beer}|\boldsymbol{x}) = 0.2401$$

Slate Recommendation (Decision Rule Optimization)

Back to Factorized Policies

$$\pi(a_1, \dots, a_K|\boldsymbol{x}) = \prod_k \pi(a_k, k|\boldsymbol{x})$$

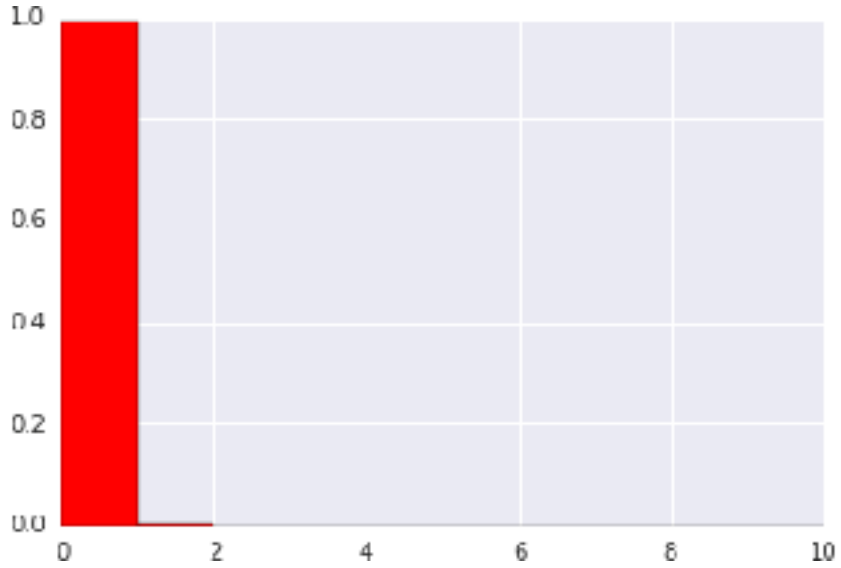
Solution: *Chen, et al. 2019* proposes a solution to this which keeps the probability spread out over the Top-K items.

CRITEO

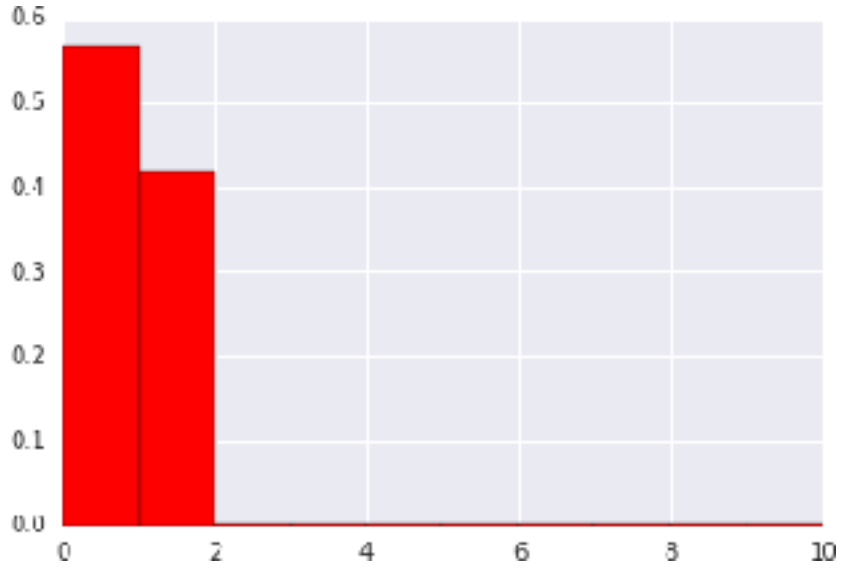
Slate Recommendation (Decision Rule Optimization)

Factorized Policies with Top-K

Vanilla IPS



IPS with Top-K (K=2)

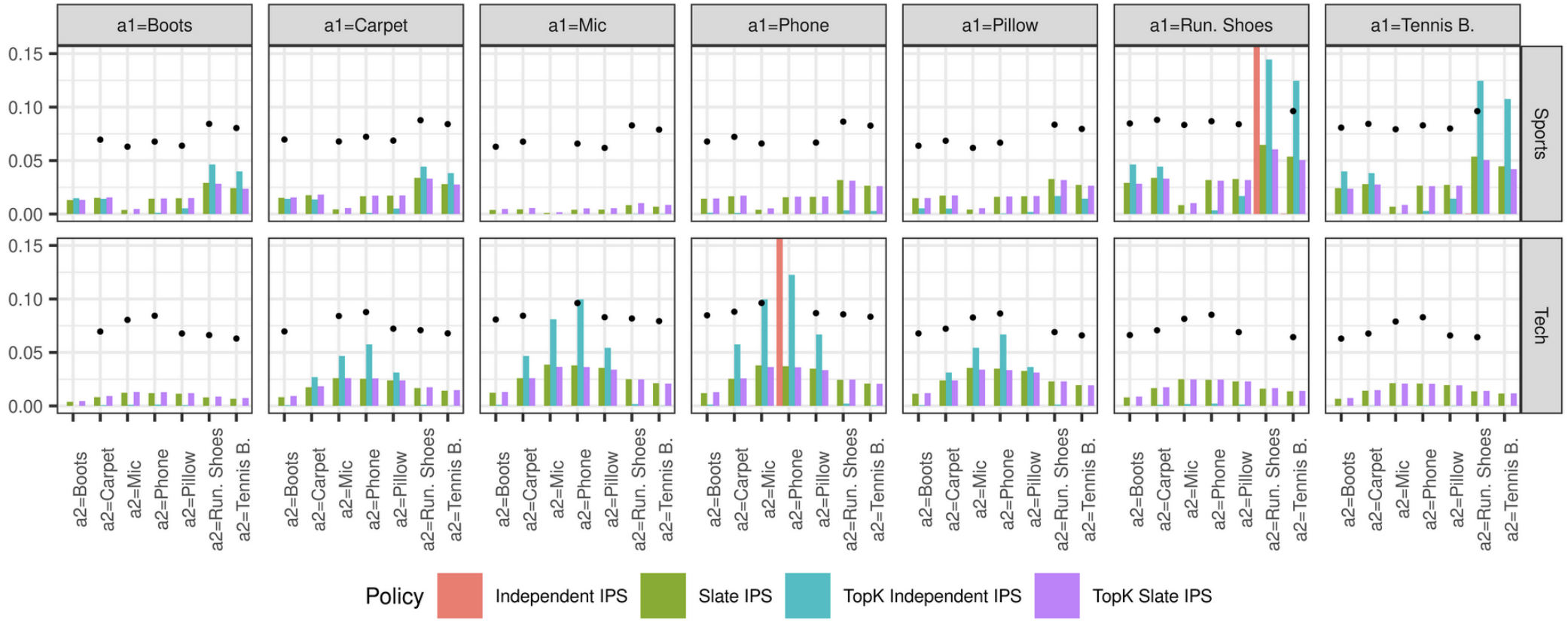


The Top-K correction modifies the loss to stop $\pi(a_k, k|\boldsymbol{x})$ converging to a degenerate distribution.

CRITEO

Slate Recommendation (Decision Rule Optimization)

Toy Example: IPS vs. IIPS



CRITEO

Slate Recommendation (Practice)

Python Tutorial 3

https://colab.research.google.com/github/otmhi/Reward-Optimizing-Reco/blob/main/ECML_Reward_Optimizing_Slate_Recommendation_with_DL_and_MIPS_Part_3.ipynb

Takeaways

CRITEO

Takeaways

- DL + MIPS is powerful - scales to large catalogues and can fit near arbitrary models.
- A/B tests produce a population split and do an excellent job of estimating reward. In contrast off-policy estimation either with DM or IPS are difficult to apply in practice:
  - Contextual bandit assumption is typically made (reward is on the recommendations not the timeline).
  - IPS: Very high variance even in idealized circumstances.
  - Practitioners fall back on proxy methods which mean that the deep learning objective and A/B test results are not aligned.

Takeaways

This tutorial surveys approaches to improving this alignment in a practical MIPS setting:

- Decision theoretic framework introduced, separating reward estimation from optimizing the decision rule.
- Methods for improving the reward estimation on full slates include IIPS and the direct method PRR.
- Some parametric models result in a MIPS compatible decision rule automatically.
- When this is not possible or desirable, optimizing the decision rule can be done using policy learning or the reinforce algorithm.
- A softmax policy is suitable for single item slates, a factorized policy can work for multiple item slates but can draw duplicates.

Takeaways

This tutorial surveys approaches to improving this alignment in a practical MIPS setting:

- Successfully achieving positive A/B tests requires:
  - Efficient reward estimation of recommendations.
  - Efficient optimization of MIPS decision rule.

References

Aouali, Imad, et al. "Probabilistic Rank and Reward: A Scalable Model for Slate Recommendation." *arXiv preprint arXiv:2208.06263* (2022).

Chen, Minmin, et al. "Top-k off-policy correction for a REINFORCE recommender system." Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining. 2019.

Beygelzimer, Alina, et al. "Contextual bandit algorithms with supervised learning guarantees." *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011.

Bottou, Léon, et al. "Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising." *Journal of Machine Learning Research* 14.11 (2013).

Ding, Qin, Hsiang-Fu Yu, and Cho-Jui Hsieh. "A fast sampling algorithm for maximum inner product search." *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019.

CRITEO

References

Dudík, Miroslav, et al. "Doubly robust policy evaluation and optimization." *Statistical Science* 29.4 (2014): 485-511.

Joachims, Thorsten, Adith Swaminathan, and Maarten De Rijke. "Deep learning with logged bandit feedback." *International Conference on Learning Representations*. 2018.

Li, Shuai, et al. "Offline evaluation of ranking policies with click models." *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018.

Malkov, Yu A., and Dmitry A. Yashunin. "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs." IEEE transactions on pattern analysis and machine intelligence 42.4 (2018): 824-836.

McInerney, James, et al. "Counterfactual evaluation of slate recommendations with sequential reward interactions." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.

CRITEO

References

Robins, James M., and Ya'acov Ritov. "Toward a curse of dimensionality appropriate (CODA) asymptotic theory for semi-parametric models." *Statistics in medicine* 16.3 (1997): 285-319.

Saito, Yuta, and Thorsten Joachims. "Counterfactual learning and evaluation for recommender systems: Foundations, implementations, and recent advances." *Fifteenth ACM Conference on Recommender Systems*. 2021.

Sakhi, Otmane, et al. “BLOB: A probabilistic model for recommendation that combines organic and bandit signals." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.

Sakhi, Otmane, David Rohde, and Alexandre Gilotte. "Fast Offline Policy Optimization for Large Scale Recommendation." arXiv preprint arXiv:2208.05327 (2022).

Swaminathan, Adith, and Thorsten Joachims. "Batch learning from logged bandit feedback through counterfactual risk minimization." *The Journal of Machine Learning Research* 16.1 (2015): 1731-1755.

CRITEO